PRIJS ƒ 7.25/Bfr. 135

Commodore INFO

ONAFHANKELIJK BLAD VOOR COMMODORE GEBRUIKERS

JAARGANG 4, NO. 4, JULI/AUG. 1987

CAD op de 128

Amiga 500

Printlist V1.1



LISTINGS

- Syntax Checksum
- Blokmonitor
- Mastermind
- Snake
- Analoge klok C-16
- Etiketten 128
- Adresbak 128
- Lijn 128
- Lichtkrant C-16

Comdex '87

Teletrade Centers

De Luxe Paint II

Parameter transport

Music Box 64

Vaste rubrieken

- Basic Miniatuurtjes
- Machinetaal
- Basic cursus
- Oud van Goudriaan
- Listings

COMMODORE-INFO
verschijnt 8 x per jaar
Jaargang 4, no. 4,
juli/augustus 1987

Uitgave:
Sala Communications

Uitgever: Drs. J. Taverne

REDAKTIE
Ir. L. Sala hoofdredacteur
J. Bodzinga adj. hoofdred.
J. Boers, R. Goudriaan,
T. v.d. Land, B. Munniksma,
drs. M. de Rooij,
drs. U. Schuurmans,
K. van der Vlies

ART DEPARTMENT
Strip: Bert Tier
Illustraties: Ben van Mierlo
Ymmot
Omslag: Ben van Mierlo

Advertentie-exploitatie:
Ing. V. Sala
R. Akker
Den Texstraat 5a
1017 XW Amsterdam
☎ 020-273198

Redaktieadres: Postbus 112
1260 AC Blaricum
☎ 02152-65695

Abonnementen en administratie: Postbus 5570
1007 AN Amsterdam
☎ 020-273198

Vragen betreffende abonnementen bij voorkeur schriftelijk, met meesturen van het omslagetiket. Telefonisch uitsluitend van 10.00 tot 15.00 uur.

Abonnement: (8 nummers)
ƒ 47,50 of Bfr. 950 per jaar
Betaling op Giro 1585491 tnv. SAC/COMMODORE-INFO Blaricum of in België op Bank BBL nr. 310050602562, vermeld SAC/COMMODORE-INFO. Oude nummers à ƒ 6,75 alleen bij vooruitbetaling op een van bovenstaande rekeningen. Ook telefonische opgave voor een abonnement is mogelijk. Bel GRATIS 06-0224222, HP-Teleservice, elke dag tot 20.30 uur (ook in het weekend). Dit nummer is alleen voor telefonische opgave van NIEUWE abonnementen.

Redactiesecretariaat:
Ron van Zalingen

Druk: Verweij, Mijdrecht
NDB, Zoeterwoude

Distributie:
in Nederland Betapress/Gilze
in België AMP/Brussel

ISSN: 0169-3085

Inhoud van dit nummer



PRINT-OUT Listing-rubriek 37

Op de CES in Chicago ontbrak Commodore geheel, op de Comdex in Atlanta was het allemaal Amiga wat de klok sloeg, maar de deelname aan de AmigaWorld in de VS werd opgeschort. Er zijn een aantal signalen dat het Commodore niet voor de wind gaat in de VS. Luc Sala rapporteert vanuit Atlanta.

Oude wijn in nauwelijks nieuwe zakken...

Comdex '87

Je vraagt je wel eens af, hoe vaak een bedrijf kan proberen de pers te bedotten. Nadat ik op een aantal computerbeurzen in de VS al tegen de PC-10 was opgelopen (voor ons al een oudje), reikte men mij op de Comdex in Atlanta weer vrolijk een persbericht uit, waarin men de PC-10 én de Amiga 500 en 2000 aankondigde.

Het nieuws is misschien, dat de PC-10 met de aangekondigde lagere prijzen nu echt serieus verkocht zou worden. Een PC-10 met één diskdrive en monitor kost nu 800 dollar in de VS, geen revolutie en nauwelijks nieuws. Misschien is het allemaal te verklaren uit het feit, dat men weer eens een vrijwel geheel vernieuwde management ploeg voor Commodore heeft laten aanrukken en de oude toppers de straat op stuurde. Rattigan, de vorige president, had het volgens de meesten helemaal niet slecht gedaan, maar Irving Gould, de oude meester van Commodore, gaf hem toch zijn congé en nam zelf de leiding op zich.

De nieuwe managers van Commodore zijn Alfred Duncan, die verantwoordelijk is voor de VS en Richard McIntyre (marketing). Er is ook weer wat geld in kas bij CBM, want de Prudential Insurance Company verstrekte een lening van 60 miljoen dollar, die eventueel in aandelen kan worden omgezet.

Amiga Software

Het is langzamerhand duidelijk, dat de 64 niet echt meer meetelt. Op de Commodore stand in Atlanta waren alleen maar Amiga's te zien. Vooral het software-aanbod was echter indrukwekkend. Er is een groot aantal, vooral kleine, softwarehuizen met heel creatieve video en audio toepassingen voor de Amiga bezig.

De grote software-bedrijven zien de Amiga meer als één van de beschikbare machines. Er is sprake van een steeds verdergaande concentratie van de software-uitgevers voor spelletjes en thuismarkt; vooral op de CES was dat heel duidelijk. Er blijven maar een paar namen overeind en nu Electronic Arts het Canadese "*Batteries included*" heeft overgenomen, is de Commodore-markt weer één van haar oudste softwarehuizen, met een bijna klassieke toewijding aan de 64-markt kwijtgeraakt.

Thuismarkt

De Amiga 500 is nu echt gericht op de thuiscomputeraar en tegen de prijs van 699 dollar in de VS is dat een moeilijke strijd. De PC-klonen zijn namelijk ongeveer even duur, maar bieden een meer universeel platform voor serieuze toepassingen. Natuurlijk is op dit moment de Amiga vrijwel onverslaanbaar op het gebied van de grafische prestaties en als recreatieve machine, maar het is de vraag hoeveel Amerikanen daar voor vallen.

Met de Amiga 2000 mikt men op een veel krachtiger segment, namelijk op de creatieve professional, die ook nog een stukje MS-DOS compatibiliteit wil hebben. Er waren een aantal desktop publishing pakketten te zien, die de capaciteiten van deze machine duidelijk beter lieten uitkomen en in vergelijking met MS-DOS machines vooral qua beeldkwaliteit en lettertypes op het scherm heel sterk overkwamen. DeskTop video is een andere toepassing, waar veel Amiga-gebruikers en ontwikkelaars zich op richten.

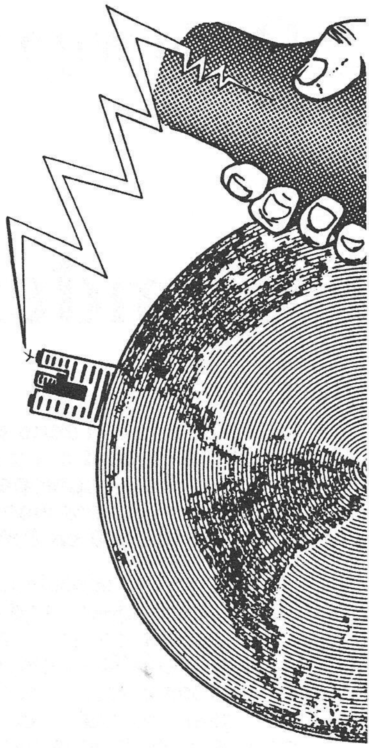
Atari

Bij het beoordelen van de toekomst van Commodore kunnen we niet om Atari heen. Daar schiet men ook niet zo erg op met nieuwe produkten. De Atari laserprinter is er bijvoorbeeld nog steeds niet en ook op de Atari PC wordt in de VS nog gewacht. Maar aan de onderkant van de markt maakte Atari een paar slimme manoeuvres. Zo heeft men de 8-bits XE plotseling in een ander jasje gestopt en noemt men het nu een Game System computer.

Dat is een verrassende aanpak, maar gezien het toch wel groeiende succes van de Nintendo spelcomputer misschien wel heel slim. Er is wel degelijk een markt voor goedkope, eenvoudig te bedienen spelcomputers, die via de speelgoedzaken verkocht kunnen worden. En voor de XE (en XL) is een overmacht aan relatief goede en goedkope software op insteekmodules beschikbaar. Dat betekent gebruiksgemak en een relatief geringer risico voor de speelgoedwinkelier, die niet zo weg is van kopieerbaar spul op cassettes of disks.

Luc Sala

Zaken doen met de PC is geen nieuws, heel wat brieven, mailing lists en klantbestanden komen uit de PC. Maar kan de computer ook gebruikt worden om bijvoorbeeld de internationale handel gemakkelijker te maken. Hier speelt telecommunicatie een grote rol, maar ook nieuwe manieren van PC gebruik. Luc Sala geeft zijn ideeën over hi-tech zakendoen, en lanceert een idee voor een hi-tech TeleTrade Center concept.



Internationale handel anno 1990

TeleTrade Centers

Zelfs in deze tijd van snelle vliegverbindingen en geavanceerde communicatietechnieken blijkt het niet zo eenvoudig, om bijvoorbeeld met een land als Japan op een eenvoudige manier zaken te doen. Er zijn diverse belemmeringen, die dit in de weg staan, maar gelukkig zijn er technologische ontwikkkelingen, die mogelijk een oplossing kunnen brengen. Een betere toegang tot informatie elders en betere communicatie met mogelijke handelspartners leidt dan vrijwel zeker tot meer handel, en dat is niet alleen in het belang van de beide zakelijke partijen, maar ook voor de hele economie.

Het is nu eenmaal voordeliger, om produkten daar te kopen, waar ze het best en goedkoopst gemaakt of ontwikkeld kunnen worden. Iemand die probeert zijn "betere" muizeval of "lekkerste" kaas te verkopen in Japan (en het volgende geldt natuurlijk min of meer voor ieder vreemd land waar men zaken wil doen), heeft het niet gemakkelijk.

Ellende

Je weet vaak niet waar te beginnen, en als je dan eindelijk mogelijke handelspartners gevonden hebt, begint de ellende pas goed. Het lijkt, zeker in het begin, of ze daar alles anders doen, welbewust niets willen begrijpen en de meest vreemde voorwaarden stellen. In de praktijk betekent dit, dat er maar heel weinig bedrijven zijn, die werkelijk zaken doen met Japan. Dat zijn echt uitzonderingen, bedrijven die door een bijzonder produkt of door vasthoudende marketing wel door de rijstebrijberg gekomen zijn.

Grote bedrijven hebben niet zoveel moeite met het benaderen van afzetmarkten in vreemde landen. Ze hebben de middelen en ook vaak de plaatselijke organisatie en contacten om dat vrij gemakkelijk te regelen. Kleinere bedrijven lopen echter snel op tegen allerlei echte of vermeende barrieres, die ze vaak doet afhaken voordat men goed en wel begonnen is. Dergelijke barrieres zijn bijvoorbeeld de taal, het gebrek aan goede contacten, de benodigde tijd en geld om potentiële handelspartners ook eens echt te kunnen ontmoeten, de administratieve rompslomp, de papierhandel en de gecompliceerde financiële afwikkeling van transacties.

En dan hebben we het nog niet over de culturele barrières, die zijn vaak net zo belemmerend, men is al snel geïmponeerd door het onbekende en ziet bergen problemen opdoemen, die er in feite niet zijn. Hoe kan nu de techniek gebruikt worden om ook de kleinere bedrijven toegang tot deze exportmarkt(en) te geven? Het is natuurlijk niet zo, dat we via de techniek het direkte persoonlijke contact, dat nog steeds zo'n belangrijke rol speelt bij vrijwel ieder commercieel contact, kunnen elimineren. Maar net als de telefoon toch een bruikbare vervanging is gebleken, kunnen ook andere technieken hier heel nuttig gebruikt worden. En laten we eerlijk zijn, voor heel wat bedrijven is een reis naar Japan gewoon niet haalbaar.

(Tele)-Communicatie

Wanneer we de problemen analyseren van een relatief klein bedrijf, dat graag zaken zou doen met bijvoorbeeld Japan, komt het aspekt communicatie steeds weer naar voren.

Aangezien het om een relatief vreemd land gaat, met een andere taal, een compleet ander schrift en natuurlijk ook andere handelsgebruiken, zou men de techniek moeten aanwenden om hier bruggen te slaan. Natuurlijk is de telefoon hier de aangewezen weg om te communiceren, maar met Japan is dat niet zo eenvoudig als het lijkt: Vrijwel niemand spreekt Engels (of durft Engels te spreken), men verbindt eindeloos door, begrijpt niet waar het om gaat, maar zegt toch steeds vriendelijk ja en dergelijke. Aangezien het tamelijk dure telefoongesprekken zijn, geeft men dan al gauw de moed op. De telefoon is pas nuttig, wanneer

men vaste contacten heeft en enige zekerheid dat men begrepen wordt. De telex is daarna de volgende stap, en zonder twijfel worden er heel wat zaken afgehandeld via dit zakelijk vrij universele medium.



Fax

In het geval van Japan wordt de rol van de telex op dit moment snel overgenomen door de facsimile-berichten. Die werken wel niet zo snel, en om ze direct te sturen is nog een heel gedoe, maar via bijvoorbeeld Faxpost is één en ander al heel goed te doen. Vrijwel alle Japanse bedrijven werken met een Fax en dat komt natuurlijk ook door het schrift. Typemachines of zelfs computers met een Japanse (Kanji of Katakana) wordprocessor zijn nog niet algemeen in gebruik. Bovendien heeft de Japanner nog een heilig ontzag voor drukletters of gezette kanji-karakters en zal eerder geneigd zijn iets met de hand op een papiertje te krabbelen om een al te formele indruk te vermijden.

Toch heeft ook facsimile nadelen, zo is het nog geen echt automatisch systeem met ontvangstbevestiging zoals de telex en verder is de terugkoppeling gering. Men weet niet, of de ontvanger het bericht wel in goede orde ontvangen heeft en of hij wel begrijpt waar het over gaat. Ik heb heel wat telexen en faxen naar Japan gestuurd, waar volstrekt onbegrijpelijke antwoorden op kwamen. Zo herinner ik me het geval, waarin op een of andere manier onduidelijk was geworden, wie kocht en wie verkocht en waar de opgegeven prijzen en hoeveelheden op sloegen, met het gevolg dat beide partijen op zoek gingen naar de gevraagde produkten. Dat soort misverstanden zijn eerder regel dan uitzondering, zowel de zender als de ontvanger van een schriftelijk bericht hebben in het algemeen moeite met het helder formuleren en interpreteren van de boodschap.
De communicatie via de telefoon, de telex of de facsimile is bij het zakendoen op zich onmisbaar, maar ook te beperkt om echt tot een sluitende en heldere uitwisseling te komen.

Electronic mail

Het versturen van ASCII tekst via een 1200/2400 bps EM-dienst werkt sneller dan een fax, maar is beperkt tot westerse lettertekens. Electronic mail heeft hier het voordeel boven een telex, dat men de boodschap wat meer vorm kan geven, grote en kleine letters en regels wit helpen om de kern van de zaak duidelijk te maken. Het is alleen jammer, dat in Japan het gebruik van electronic mail niet zo erg gemakkelijk gaat, men heeft een speciaal toegangsnummer van de KDD (PTT) nodig en ook hier loopt men tegen het gebrek aan computers op. Wat betreft E-mail ligt men nog een stuk achter op bijvoorbeeld de VS, al wordt er de laatste tijd wel veel meer aan gedaan. Er zijn nu goedkope modems op de markt en er komen allerlei informatie-diensten van de grond. In dat opzicht loopt men echter, net als in de Nederlandse situatie met Viditel, tegen een vrij onduidelijk tweesporen beleid aan.

Communicatiebarrières maken de internationale markt voor kleine bedrijven nog steeds ontoegankelijk

In wezen zijn half-duplex communicatiesystemen met een ingebouwde vertraging niet optimaal, en dat geldt zowel voor telex als facsimile of electronic mail. De telefoon is in dat opzicht veel beter, maar de taalbarriere is hier de spelbreker.

Nieuwe wegen

Bovenstaande comunicatiemedia zijn op dit moment vrij eenvoudig toegankelijk, ook voor kleinere bedrijven. Natuurlijk samen met de aloude brief die tegenwoordig niet alleen via de post, maar ook via allerhande koeriersdiensten verstuurd kan worden.

Wanneer we nadenken over betere communicatiemethoden, zou een combinatie van de telefoon en de facsimile natuurlijk heel handig zijn om tegelijkertijd iets op papier over te kunnen sturen en er dan ook over te praten. Iets dergelijks kan ook met electronic mail en heet dan "**Voice over Data**", maar de PTT is er, zeker in het internationale verkeer, niet al te blij mee. Men heeft namelijk het telefoonverkeer al zo geoptimaliseerd, dat er in de praktijk alleen maar een "echte" verbinding is, wanneer men ook werkelijk spreekt. Door allerlei "multiplexing" en half-duplex technieken wordt een telefoonlijn tijdens een normaal gesprek dan ook maar voor 1/3 tot 1/5 van de tijd echt geopend. Wanneer daar nu een data-signaal van een modem of faxs, die met continu "carrier" signalen werken, bij komt, wordt zo'n telefoongesprek eigenlijk te duur (onrendabel) voor de PTT. Men is er dus niet erg voor om dit soort oplossingen,enenen die technisch heel goed haalbaar zijn, toe te staan of actief te bevorderen. "Voice over Faxs" is echter vanuit communicatie-oogpunt bekeken een grote stap vooruit, zeker voor bedrijfsmatige toepassingen.

Videophone

De beeldtelefoon is als techniek al meer dan tien jaar oud, en Philips heeft in het verleden al meermalen apparatuur laten zien, waarmee dat technisch heel goed mogelijk was. De benodigde verbindingen waren echter te breedbandig, er was een aparte verbindingskabel of satellietkanaal voor nodig of als alternatief een combinatie van een heleboel telefoonlijnen. Daarmee bleven de kosten van een beeldtelefoongesprek te hoog en is het medium nooit echt van de grond gekomen. De laatste jaren zijn er echter coderingstechnieken ontwikkeld, die het mogelijk maken om met veel minder bandbreedte toch een bruikbare beeldverbinding op te bouwen. Met een beperkte digitalisatie en gebruik van "still-video" (stilstaande videobeeldjes) kan men zelfs met een gewone telefoonverbinding nu al beperkt aan beeldtelefonie doen. Er zijn diverse video-phone systemen op de markt, o.a. de Lumaphone van Mitsubishi. Met de geleidelijke groei naar ISDN verbindingen,

waarbij men 64 kbps verbindingen of veelvouden daarvan tot zijn beschikking krijgt, zal in dit opzicht zeker nog veel gaan gebeuren. Een videofoon is met name in het zakelijk verkeer, waar men niet alleen de partner in de ogen wil kijken, maar ook zijn produkten wel eens wil zien, een belangrijk instrument.

Kleurenfacs

En wanneer we dan aan afbeeldingen van produkten denken, dan is de stap naar kleur natuurlijk een kwestie van tijd. Het is op dit moment technisch mogelijk om een kleurenfacsimile systeem te maken, in de praktijk heb ik dat echter nog nergens als commercieel produkt gezien. De overtuigingskracht van een kleurenplaatje is echter vaak zoveel beter dan van een zwart-wit afbeelding, dat dit vast niet lang op zich laat wachten.

PC toepassingen

Behalve de telecommunicatie, de netwerken, de koppeling van computers middels WAN's (Wide Area Networks) biedt de computer natuurlijk ook een hele reeks andere mogelijkheden om commercieel gebruikt te worden. Neem bijvoorbeeld het gebruik als vertaalcomputer. op dit moment zijn er een aantal grote mainframe-systemen, waarop men teksten kan vertalen van en naar het Japans. Op een recente tentoonstelling in Tokio zag ik nu echter ook programma's, die datzelfde op een relatief kleine PC deden. Het zal taaltechnisch nog wel allemaal niet 100% verantwoord zijn, maar wanneer je links de tekst in het Engels hebt en rechts de Japanse karakters verschijnen, is dat meer dan een stapje in de goede richting. Natuurlijk liever een perfecte vertaling, maar dit is al veel beter dan altijd met een tolk te moeten werken en de ontwikkeling staat zeker niet stil.

De computer kan ook, middels koppeling aan geschikte on-line databanken of met behulp van "ingevroren" databanken op bijvoorbeeld CD-ROM, gebruikt worden om informatie op te vissen. Juist de ontwikkeling van Expert-systemen en "*Artificial Intelligence*" technieken zou bij het complexe terrein van de internationale handel en de duizenden regels, voorschriften, formulieren en dergelijke van onschatbaar nut kunnen zijn. De moderne laserprinters zijn dan ook in staat, om de overstap tussen onze letters en de vreemde karakters zonder problemen te maken.

Creatieve technologie

In het verlengde van wat ik hierboven heb aangegeven, zijn er natuurlijk nog eindeloos veel andere technieken om het moeilijke terrein van de interculturele communicatie, en daar praten we toch over wanneer we het hebben over handel met bijvoorbeeld Japan, open te breken. De computers worden zo krachtig, dat spraaksynthese, geavanceerde grafische afbeeldingen, patroonherkenning en wat al niet mogelijk worden.

Waar het echter om gaat, is om goede toepassingen te vinden voor de nieuwe technieken en om aan te tonen, dat ze ook praktisch bruikbaar zijn. In verband met de internationale handelspromotie geloof ik, dat daar een uniek en potentieel ook zeer lucratief terrein is, om deze technieken toe te passen. Het probleem is alleen, dat het realiseren van zo'n toepassing zich wel uitstrekt over de halve wereld en dat juist de problemen die er mee opgelost zouden kunnen worden, ook de barriere vormen om tot praktische toepassing te komen.

Voorbeeld-centrum

Wanneer er nu, misschien met steun van de respectievelijke overheden en/of handelsbevorderende organisaties, een soort voorbeeldcentrum zou kunnen komen om de levensvatbaarheid van dit soort technologische oplossingen te demonstreren en aan te tonen, lijkt mij dat een positieve zaak. Daarbij denk ik aan het opzetten van tweeling-centra in bijvoorbeeld Amsterdam en Tokyo, waar men de moderne communicatie en informatie-technologie gebruikt om juist de contacten tussen kleinere bedrijven en organisaties in de twee landen te bevorderen. Zoiets ligt in het verlengde van de funktie van bijvoorbeeld een Trade-Center, maar heeft als extra de telecommunicatiefaciliteiten, die men tegenwoordig graag in de vorm van het "Teleport" concept wil concentreren in bepaalde plaatsen.

Het TeleTrade Center kan de ideale combinatie van Trade Center en Teleport vormgeven.

Zo'n *TeleTrade*-handelscentrum voor internationale contakten is een combinatie van mogelijkheden, die ook in dit concept genoemd zijn. Telecommunicatie, Tradepromotion, met het woord **TeleTrade Center** is dat allemaal onder één noemer gebracht.

Met een soortgelijk centrum in beide landen, die gekoppeld zijn via allerlei telecommunicatieverbindingen en met de technische middelen om bijvoorbeeld documentatie en promotiemateriaal te maken, krijgt handelspromotie een heel nieuw gezicht. Vooral voor kleinere bedrijven is er dan de mogelijkheid, om veel sneller en directer contact te maken met potentiële handelspartners, zonder direct in het vliegtuig te hoeven stappen.

En laten we eerlijk zijn, we praten hier niet over technische wonderdingen, maar over apparatuur die hetzij commercieel verkrijgbaar is, hetzij in een beta-test fase verkeert en met wat goede wil best beschikbaar gesteld kan worden door de bedrijven, die het ontwikkelen. Dat geldt overigens in nog sterkere mate voor de software en databank-faciliteiten, die een integraal deel van zo'n centrum zouden moeten vormen. Zeker in het begin is voorbeeld-functie van zo'n centrum en de publiciteit, die het zeker zal krijgen, een krachtig middel om bedrijven tot medewerking te verlokken.

Realisatie

Is het nu mogelijk om zo'n centrum ook werkelijk te realiseren? Als journalist kun je natuurlijk wel allerlei ideeëen lanceren, maar voor de realisatie ervan zijn meer dingen nodig dan wat artikelen in een blad. Gelukkig zijn er in ons land genoeg organisaties, die met een idee als het bovenstaande iets kunnen doen.

Hoewel het natuurlijk wat vreemd aandoet, om te spreken over een traditionele functie van ons land wanneer we het hebben over computers en telecommunicatie, valt het idee van een TeleTrade Center toch heel duidelijk binnen de Japans-Nederlandse handelstraditie. Want het waren de Hollanders, die middels een vooruitgeschoven handelspost (Decima) eeuwenlang fungeerden als het transmissiemedium tussen het Japanse keizerrijk en de Westerse wereld. En hoewel de technieken natuurlijk veranderen, zou het realiseren van de hierboven beschreven TeleTrade Centers ook nu een dergelijke functie hebben. En daarmee zou de rol van ons land als handelsnatie ook in de toekomst aan inhoud en betekenis kunnen winnen.

Luc Sala

Door alle aandacht voor de MS-DOS compatibele Amiga 2000 is de Amiga 500 wat op de achtergrond geraakt. Niettemin blijft de Amiga 500 een belangrijke hobbymachine met 68000-kracht voor de gewone man. Waar anders vindt men een 16 bits-machine onder de *f* 1400,- met zoveel mogelijkheden als de 500? Wij voelden deze kleinste Amiga eens nader aan de tand.

68000-kracht voor de gewone man

Amiga 500



Computerhobbyisten willen steeds meer. Verwend door fraaie videobeelden, flitsende animaties en snelle processoren nemen zij geen genoegen meer met eenvoudige 8-bits CPU's en ruwe graphics met strompelende sprites. De oorspronkelijke Amiga 1000 trok dan ook veel bekijks tijdens demonstraties en menigeen voelde de begeerte bij zich opkomen. Helaas, de prijstelling ging ver boven het hobbybudget uit en wie niet jaren lang water en brood wilde eten moest nog maar wat geduld hebben met een C-64 of C-128. Gelukkig werd het wachten beloond en introduceerde Commodore onlangs de Amiga 500 als gedoodverfde opvolger van de C-64/128 lijn van 8-bitters. Of deze verwachtingen ten aanzien van de hobbycomputer van de jaren 90 zullen uitkomen moet echter nog blijken.

Het is nog maar enkele jaren geleden dat Commodore de C-128 als opvolger van de succesvolste homecomputer aller tijden, de C-64, aankondigde. Mikkend op de grote hoeveelheid beschikbare software dacht CBM met een nieuwe voor 99,9% met de C-64 compatibele hobbycomputer het succes van de oude topper te kunnen evenaren. Hoewel de verkoop van de C-128 niet geheel onverdienstelijk bleek vielen de verkoopresultaten tegen. Vervolgens stak men de oude 64 in een flitsend jasje (de C-64 II) en ondersteunde het Mac-achtige GEOS userinterface. Ook kwamen er krachtiger diskdrives en zelfs een harddisk beschikbaar.
Al deze welkome vernieuwingen veranderden natuurlijk niets aan de grenzen der 8-bitsmogelijkheden. De "final frontier" is nu wel zo ongeveer bereikt en de concurrentie in de vorm van MS-DOS PC-klonen en 68000-machines rukte langzaam maar zeker op. Nog steeds hebben de C-64 en 128 een flinke aantrekkingskracht op de (beginnende) hobbyist, onderwijs en kleine zakelijke gebruiker. De lage prijs en de enorme hoeveelheid software wegen voor talloze gebruikers ruimschoots op tegen de nadelen van trage rekenkracht, mindere grafische kwaliteit en beperkte geheugencapaciteit. Toch groeien onvermijdelijk steeds meer serieuze hobbyisten boven de mogelijkheden van de C-64 en C-128 uit. Wie echter meer wil zal ook meer

moeten betalen en veel financiële armslag heeft de doorsnee Nederlandse hobbycomputeraar niet.
Met de introductie van de scherp geprijsde Amiga 500 hoopt Commodore deze kategorie hobbyisten over te halen om de stap naar een 68000-systeem te maken. De Amiga 500 is met zijn ingebouwde diskdrive en bijgeleverde muis eigenlijk nauwelijks duurder dan een C-64 II met los diskettestation en biedt alle mogelijkheden van de nieuwe generatie homecomputers. Daartegenover staat, althans in Nederland, nog het nadeel van de traag op gang komende softwarestroom. Maar laten wij eerst eens kijken wat de Amiga 500 allemaal te bieden heeft.

De basisconfiguratie

In tegenstelling tot de Amiga 2000 is de 500 geen bouwdoos (een open systeemarchitectuur), maar een vrijwel gesloten, compleet systeem. In de op de C-128 lijkende systeemkast met ingebouwde 3.5 inch diskdrive is geen ruimte voor inbouwkaarten of extra drives. Hoogstens kan aan de onderzijde nog een geheugen-uitbreidingskaart met klok ingestoken worden. Alle verdere uitbreidingen moeten via de expansion port aan de linkerzijde van de computer.
De Amiga 500 biedt de serieuze hobbyist meteen alles wat voor jarenlang probleemloos computeren en doorgroeien noodzakelijk is. Het ingebouwde 880 KB 3.5 inch diskettestation kan qua datacapaciteit de meeste spelletjes en programmatuur ruimschoots aan. Wie meer mogelijkheden of gebruikersgemak ambieert kan via de floppy-poort nog een extra drive aansluiten.
Standaard zijn een 256KB metend ROM, met het operating system Kickstart 1.2, en een 512KB groot RAM aan boord. Desgewenst kan de geheugencapaciteit tot 1MB opgevoerd worden. Ruimte genoeg dus voor grote spreadsheed-modellen en gedetailleerde graphics of videobeelden.
Het hart van de Amiga wordt gevormd door de 32/16-bits Motorola MC 68000 CPU met een kloksnelheid van 7,16 MHz. De interne (adressbus) dataoverdracht is 32-bits. Die overdracht per systeembus verloopt in woorden van 16 bits. Verder beschikt de MC 68000 over een aanzienlijk krachtiger on board instructieset dan de 6502 CPU's van de C-64 en C-128. Dat bespaart veel tijd bij het programmeren en het uitvoeren van opdrachten. Bovendien behoort multitasking tot de mogelijkheden.
Net als bij de Amiga 1000 en 2000 wordt de 68000 CPU ondersteund door drie coprocessoren:

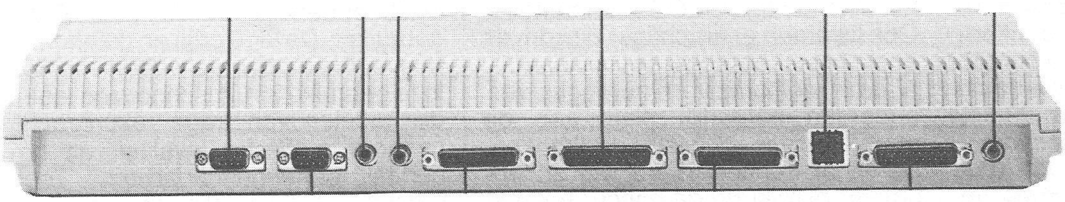
- De videochip **Denise** kent vier verschillende grafische schermen: 320 x 256, 320 x 512, 640 x 256 en 640 x 512 pixels. Het 50 Hz videobeeld heeft een oplossend vermogen van 625 lijnen en een geheugen van maximaal 512 KB. In de tekstmodus zijn naar keuze 60 of 80 tekens beschikbaar met maximaal 25 gekleurde regels. Het kleurenpalet biedt 32 kleuren (bij 320 beeldlijnen) of 16 kleuren (bij 640 lijnen) met maximaal 4096 mengkleuren. Verder behandelt Denise de spritebesturing en botsingsdetectie.
- De poort-chip **Paula** behandelt de DMA-besturing van de seriële poort, parallelle poort, controllerpoort, toetsenbord en audio-I/O.
- De Bit-Blitter- of animatie-chip **Agnus**. In de Amiga 500 is sprake van een speciale custom uitvoering, de zogenaamde Fat Agnus, waarbij nog meer functies in één en dezelfde IC zijn ondergebracht. Deze coprocessor verzorgt in nauwe samenwerking met de beide anderen de vloeiende Amiga-animaties.

De interfaces

Eindelijk lijkt Commodore haar incompatibiliteitspolitiek te laten varen. In de hobbydagen van weleer was het aansluiten van niet-Commodore hardware op een C-64 of C-128 een crime. Eigenzinnige connectoren vroegen om specialistische en dure verloopkabeltjes of vele uurtjes zweten met de soldeerbout. Ook de parallel Centronics printerpoort van de Amiga 1000 leed aan dit euvel.

Bij de Amiga 500 zijn de poortaansluitingen echter allemaal PC-standaard. De **RSC 232-uitgang** laat zich moeiteloos aansluiten op tal van modems, afdrukkers en communicatie-kabeltjes. Alleen voor MIDI-toepassingen is een speciale adapterkabel nodig. De Baudrate is programmeerbaar tot maximaal 31.250 baud en het transport van ASCII-files mag met geschikte communicatiesoftware dus geen problemen meer opleveren. Datzelfde geldt eveneens voor de **parallelle Centronics-poort**. Een speciale adapterkabel is niet meer nodig.
Een **tweede parallel-interface** verzorgt de communicatie met ten hoogste drie externe drives. Er is keuze uit 3.5 en 5.25 inch floppy drives en een hard disk met ingebouwde controller van Commodore zelf of onafhankelijke merken. De Amiga 500 kan maximaal 1 extern floppystation voeden. Meerdere floppy drives of een hard disk zullen hun eigen stroomvoorziening moeten meebrengen!

Een **monitor** kan zowel via een tulpkabel (BAS monochrome) als een 25-pins RGB A/D-plug aangesloten worden. Wie geen geld voor een kleurenmonitor heeft kan dus nog vooruit met zijn of haar oude monochrome model. Bezitters van een A1081 kunnen al direkt van de fraai gekleurde HIRES graphics genieten.



INTERFACES

van links naar rechts:
poort 1 voor muis, joystick
poort 2
stereo/audio uitgangen
aansluiting externe diskdrive
seriële RS232 poort
parallel Centronics poort
aansluitbus extern netwerk
monitoraansluiting
monochrome-video (BAS) aansluiting

Jammer genoeg ontbreekt wel een RF-modulator. Deze zijn natuurlijk ook los te koop, maar dat betekent weer een extra uitgave, een ontsierend kastje op je bureau en een mindere beeldkwaliteit.

De geluidsprestaties van de Amiga 500 komen via een interne luidspreker nimmer tot hun recht. Aansluiten op de stereo brengt daarentegen menige moderne muziekliefhebber in verrukking. De tulpbussen stereo-in en -out maken deze verbinding tot het simpelweg insteken van vier cinch-plugjes.

De standaard 9-pins **muis/joystickpoorten** zitten aan de achterzijde. Minder handig dan de meer gebruikelijke plaatsing aan de zij- of voorkant, maar de Amiga 500 zit nu eenmaal stampvol met aan de rechterkant de 3.5 inch floppy en links de expansion port. Behalve een muis en joystick kunnen op de beide poorten ook een joyball, lichtpen of grafisch tekenbord aangesloten worden.

De **expansion port**, in feite een processorbus, biedt de veeleisende gebruiker nog de mogelijkheid om meer RAM-geheugen, een extra coprocessor, een harddisk, meet/regel-relais, robot-controllers of andere randapparatuur te installeren. Vreemd is dat bij de eigen MS-DOS-emulator **Sidecar** een aansluitprobleem dreigt te ontstaan. De systeembus zit aan de verkeerde kant en is nog 180 graden gedraaid ook. Een adapterkabeltje brengt uitkomst, maar het geheel wordt er niet fraaier op.

Tussen de poorten aan de achterzijde van de Amiga 500 treft u ook de netstroomconnector aan. Net als bij de oudere Commodore-modellen is voor een losse netvoeding gekozen. Nette inbouw was waarschijnlijk, gezien het ruimteprobleem, onmogelijk. Een ander kritiekpuntje is dat de aan/uitschakelaar op het trafokastje zit. Verre van handig daar de meest gebruikers deze lelijke voedingseenheid ergens onder hun bureau zullen wegmoffelen.

In de Amiga 500

Na het loschroeven en oplichten van de bovenkap valt het inwendige van de Amiga 500 te bewonderen. Zoals gezegd is alle hardware, behalve het diskettestation, op een hoofdkaart gemonteerd. Uiterst links op deze kaart zit de MC 68000 met vlak daarnaast de expansion port. Eventuele uitbreidingen hebben daarom vrijwel direkt kontakt met de CPU.

Direkt rechts van de 68000 processor staat het 256KB ROM met de ingebakken Kickstart versie 1.2. Om dit ROM zijn de drie coprocessoren gegroepeerd waarbij de dikke zwarte Agnes duidelijk opvalt. Nieuw is de custom Gate-Array-Chip **Garry** die snel en nauwkeurig tal van logische functies, o.a. het aansturen van TTL-monitoren, vervult. De Amiga 500 is tot nu toe de enige van de Commodore 68000-familie die over deze coprocessor beschikt.

Links aan de voorzijde zit de standaard geheugenbank met 512KB aan vrij RAM. Deze bank bestaat uit 16 256KB RAM-chips met een accestijd van 150 nanoseconden. Aan de rechterzijde van deze RAM-bank kan de uitbreidingskaart met 512KB extra RAM, klok en accu in de daarvoor bestemde connector gestoken worden. Deze ruimte is via de bodemplaat toegankelijk zodat de gebruiker de 500 niet behoeft open te schroeven.

Verder zitten er nog een quarts video-oscillator met een frequentie van 28.63 MHz (dicteert de 68000 een kloksnelheid van 7.16 MHz) en enkele DMA I/O-chips op de hoofdkaart. Veel meer IC's zijn er niet. Kortom een keurig staaltje van geïntegreerde chiptechniek.

De ingebouwde 3.5 inch floppy disk drive biedt een comfortabele 880KB aan geformatteerde opslagcapaciteit. Dat is altijd nog 160KB meer dan de in concurrent Atari ST gebouwde drive. Daar de interne diskdrive zijn voeding uit de Amiga 500 trekt konden de afmetingen bescheiden gehouden worden. Onze drive functioneerde zonder mankeren (een verademing na al die 1541-kuren) en viel zelfs redelijk snel te noemen.



Het toetsenbord

Het met de systeemkast geïntegreerde keyboard ziet er met zijn strakke layout en 96 toetsen profes sioneel uit. Het geheel lijkt veel op een kruising tussen een C-128 en een Atari 1040 ST. De huiscomputer-mode van het einde van de jaren tachtig.

De layout toont een duidelijke driedeling. Circa tweederde deel van het schuine front der systeemkast wordt in beslag genomen door een comfortabel QWERTY-toetsenbord. Uiterst rechts zit een numeriek toetsenbordje met Ins-, Del-, Home-, End-, PrtSc-, NumLock-, PgUp-, PgDn- en Scroll Lock-toetsen. Tussen het numerieke en QWERTY-bord zitten nog een driehoekig (of een omgekeerde T) cursorblok en de toetsen Del en Help. Boven het QWERTY-deel vindt de gebruiker uiterst links de Esc-toets en rechts daarvan de functietoetsen F1 t/m F10.

De beide Amiga(**A**)-toetsen zijn gebleven. Een echte resetknop ontbreekt nog steeds zodat voor het resetten de combinatie A-A-Ctrl gedrukt moet worden.

Het schrijfmachine-bord geeft nauwelijks aanleiding tot klagen. De aanslag is voldoende trefzekker, de Enter- en shifttoetsen zitten waar de typer hen verwacht. Over de plaats van de Esc-key valt de twisten. Het is maar waar je hem gewend bent. De cursor-driehoek vonden wij, aan PC-

keyboarden gewend, niet zo handig. Maar een Amiga wordt per slot van rekening hoofdzakelijk met de muis bediend. Alle toetsen zijn volledig processorgestuurd en in de praktijk zullen zelden uitleesfouten optreden.

De muis

Alle Amiga's hebben een staartje met een muis er aan. Het digitale Amiga knaagdier is een eenvoudig optomechanisch model met twee be-dieningsknoppen. Een kunststoffen bal brengt de bewegingen op het bureau aan de sensoren over. Voor het noodzakelijke reinigen zit er aan de onderzijde een handig klapluikje.
De muis is tamelijk fors en vergt behoorlijk wat tafel- en elleboogruimte. Na even wennen wordt de 500-muis echter tot een betrouwbaar en plezierig aanwijsgereedschap. Voor games en het maken van tekeningen prefereren wij echter respectievelijk een joystick en een lichtpen.

Graphics en animaties

Wat graphics en animaties op huis- en personal computers betreft staat de de Amiga nog steeds aan de top. Zelfs professionele videofilmers en reklamebureau's maken gebruik van de betaalbare Amiga-mogelijkheden. Pakketten als DeLuxe Paint II Aegis animator en DeLuxe Video behoren tot de bestverkochte 68000 software. Volgens de opiniepeilingen beschikt meer dan de helft van de Amiga-bezitters over een teken- en/of animatiepakket.
De grafische beeldvorming is bij de 500 (en andere Amiga's) grotendeels uitbesteed aan de coprocessor Denise. Deze Display Encoder neemt het tijdconsumerende rekenwerk van de 68000 CPU over. Zoals bekend mengt Denise de drie basiskleuren rood, groen en blauw in 16 verschillende kleurtrappen tot een palet van 4096 mengtinten. Het aantal beschikbare kleuren hangt af van de schermresolutie. Kiest de gebruiker voor 32 verschillende kleuren dan bedraagt het oplossendvermogen 320 x 256 pixels (LORES), bij 16 kleuren 640 x 256 pixels (HIRES). Een pixelscherm heet bij de Amiga een **bit mapped plane**, of in gewoon Nederlands een grafische geheugenkaart. Bij elk punt op de monitor hoort een grafische geheugenlokatie met bijpassend kleurenregister.
Denise kan vijf LORES planes tegelijk besturen. In de hogere 640 x 256 pixels resolutie kan de grafische chip maximaal vier planes aansturen. Hoe hoger de schermresolutie des te minder kleuren en planes er beschikbaar zijn. Al te kleurrijke beelden gaan achteruit in videokwaliteit en dat kan storend werken bij het afwisselen tussen of mengen van conventionele en computer-videobeelden.
Een tweede functie van Denise is het instellen van de **interlace modus**. Op zich een simpel gebeuren waarbij de grafische chip de beeldfrequentie van de horizontale beeldlijnen verandert. Er is keuze uit 50 en 25Hz. Deze switchmethode heeft grote gevolgen voor de videodisplay. Bij 50Hz neemt het oog geen enkele beeldflikkering meer waar. Animaties verlopen daarom geheel vloeiend. Een nadeel is de resolutie die bij 50Hz maximaal 320 x 512 pixels meet. Een lager oplossend vermogen is voor snelle animaties geen groot bezwaar. Wel voor statische graphics, Denise schakelt dan in de 25Hz modus met een maximale resolutie van 640 x 512 pixels. Bij 25Mz neemt het beeldflikkeren uiteraard toe.
Kleurovergangen regelt Denise via de **Hold and Modify-modus** (HAM). De vloeiende kleurovergangen ontstaan doordat de grafische coprocessor de intensiteit van één der drie basiskleuren in de opeenvolgende pixels verandert. In feite wordt dus punt voor punt gewijzigd, maar dat gaat zo snel dat het oog een wervelende kleurbeweging waarneemt. Hold and Modify werkt alleen in de LORES- of Interlace-modus.
Alle actie vindt plaats op 16-kleurige speelvelden, de **Play-Fields**. Van zo'n speelveld is doorgaans slechts een vensterdeel zichtbaar. Het werkelijke Play-Field is veel groter dan de monitor kan laten zien. Denise verstaat de kunst om twee speelvelden, nu met elk slechts 8 kleuren, over elkaar te schuiven, in twee richtingen HIRES te scrollen en zelfs van grootte en vorm te veranderen.
Net als bij de C-128 en C-64 zijn er natuurlijk ook sprites voorhanden. Denise bestuurt maximaal acht sprites in vier kleuren. Een sprite is maximaal 16 punten breed. De vertikale lengte is onbeperkt. Verder behandelt Denise de sprite collisions en het stellen van sprite-prioriteiten.
Ook Fat Agnus, de DMA adresgenerator, is onmisbaar voor het creëren van de flitsende Amiga animaties. Deze dikzak kan in de Amiga 500 bliksemsnel 1MB aan videodata verversen. Direct Memory Access omzeilt de hoofprocessor en spreekt het geheugen dus direkt aan.
De Amiga 500 en 2000 kennen twee verschillende soorten RAMs. Memory boards die de 68000-bus gebruiken bieden het zogenaamde **fast memory**. Fast wil zeggen dat het RAM de bus-access niet met de video Blitterchip behoeft te delen. Dat was wel het geval op de Amiga 1000 waarbij de Blitter, dankzij zijn hogere systeemprioriteit, buscycli stal van het interne **chip memory,** hetgeen de programma-uitvoer vertraagde.
Op de Amiga 500 met geheugenuitbreiding bestaat de helft van het 1 MB grote RAM (dus 512K) uit fast- en de andere helft uit chip memory. De Blitter kan daarbij rustig te keer gaan met het chip-video-geheugen terwijl de software aanzienlijk sneller in het fast Ram draait.
Agnes bestaat uit drie onderdelen. De **DMA-controller** hebben wij zojuist al besproken. Het belangrijkste onderdeel van Agnus is echter de coprocesor **Copper**. Copper wordt in feite geheel door de electronenbundel van de monitor gestuurd en schrijft de data van en naar de registers. Er zijn drie bevelen mogelijk: MOVE verplaatst de data, WAIT doet de coprocessor wachten tot de electronenbundel de gewenste schermplaats bereikt heeft en SKIP is er voor het overslaan van bevelen. Copper verandert slechts de registers en grijpt niet zelf op het RAM aan.
Het derde IC-onderdeel van Agnes is de zogenaamde **Blitter**. De Blitter is een grootschalige verhuizer van grafische datablokken. Commodore zelf geeft 1.000.000 pixels en 60 beelden per seconde op. Interessant is de animatie-optie, waarmee maximaal drie bewegende elementen met elkaar kunnen worden verbonden. Bijvoorbeeld de benen en het hoofd van een kartoonfiguurtje animeren.

Home-Video

Alle Amiga animaties, speelschermen en grafische creaties kunnen via de video-output op de eigen videorecorder worden overgenomen. Daar is echter meer voor nodig dan de reklame doet geloven. De benodigde video- en grafische software wordt niet meegeleverd en dergelijke pakketten kosten helaas nog altijd enkele honderden guldens. Ook is er doorgaans een speciaal verloopkabeltje nodig om de Amiga 500 video-uitgang met de VCR of montage-unit te kunnen verbinden.
Voor het combineren van een computer en een VCR- of camerasignaal is een zogenaamde **genlocker** nodig. Superpositie (over elkaar heen) van beelden is alleen mogelijk als de ene videobron synchroon en de andere asynchroon is. De genlocker regelt de beide signalen op elkaar af. Commodore levert de **Genlock 1300** voor PAL-videosystemen, maar in Duitsland en bij gespecialiseerde firma's zijn ook apparaten van andere mer-

ken te koop. De Genlock 1300 wordt tussen de RGB-poort en VCR geschakeld en stemt het composiet videosignaal, de stereo-audiolijnen en het Amiga computersignaal op elkaar af. De prijs van deze Commodore Genlock ligt rond de *f* 600,-.
Voor het digitaliseren en creatief bewerken van videobeelden is weer een aparte digitizer nodig. **Digi-View** van Newtek biedt de Amiga-bezitter een compleet HAM video-digitizing systeem in 4096 kleuren. Kleurenvideobeelden kunnen met Digi-View gedigitaliseerd en vervolgens met Deluxe Paint II, DeLuxe Video of het eigen Digi-Paint bewerkt worden. Allemaal prachtige mogelijkheden voor de videohobbyist, maar helaas niet goedkoop. Vooral als men de Amiga ook nog eens met een videoprocessor of special effects generator (SEG) wil kombineren.

Softwaremarkt.

Tot op heden is een Amiga-softwaremarkt in Nederland nog steeds niet van de grond gekomen. De één geeft de schuld aan de welig tierende piraterij. De ander aan de ongeïnteresseerde (?) dealers die te hoge prijzen zouden berekenen.
Zoals gewoonlijk ligt de waarheid waarschijnlijk ergens in het midden. Het aantal verkrijgbare pakketten blijft echter bedroevend en de prijzen gaan menige hobbybeurs ver te boven. Dat is geheel anders over onze oostgrens waar de keuze groot en de prijzen redelijk zijn.
Hopelijk doet de snel stijgende populariteit van de Amiga ook in Nederland het tij keren.

De minpuntjes

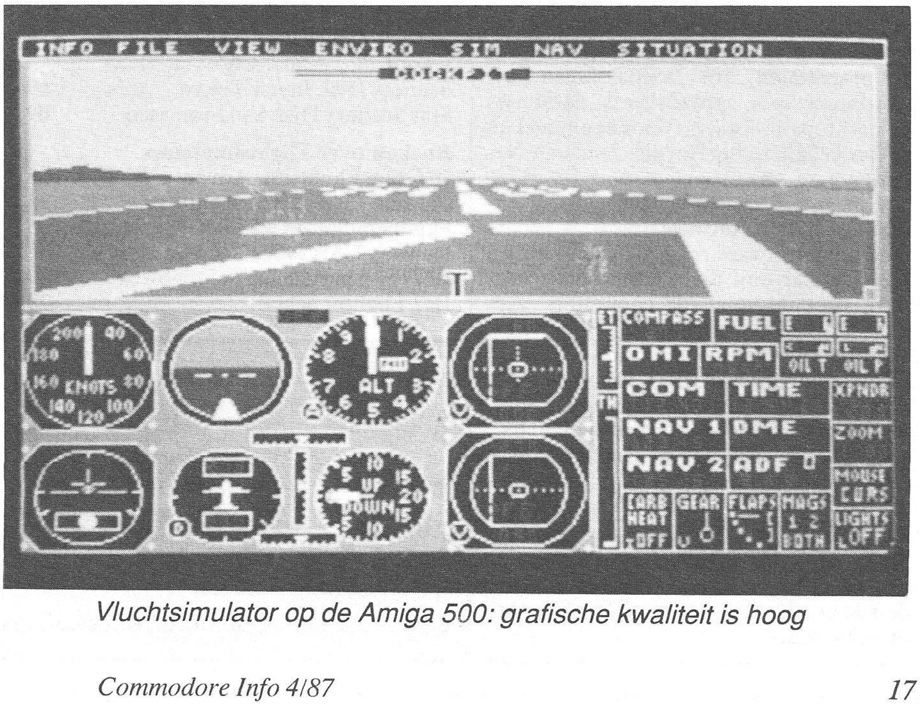
De lage prijs maakt kritiek op de Amiga 500 moeilijk. De hieronder vermelde minpuntjes betreffen dan ook voornamelijk waarschuwingen omtrent de grenzen van het systeem en bijkomende kosten:

- De **gesloten systeemarchitectuur** belemmert grote uitbreidingen. Een expansiepoort is zò vol en in de kast is geen plaats meer voor de inbouw van extra kaarten of drives. Verder bemoeilijkt de compacte bouw de service en doe-het-zelf soldeerwerk.
- De **incompatibiliteit** met andere gangbare computersystemen zoals de C-64/C-128 en MS-DOS machines. Hobbyisten met een kast vol Commodore software kunnen die bij de aanschaf van een Amiga 500 meteen van de hand doen.
- Een echte **RESET-knop** ontbreekt. De A-A-Ctrl-combinatie werkt minder handig.
- De **externe voeding** oogt minder fraai en een aan/uit- schakelaar hierop is vaak moeilijk bereikbaar.
- De **MS-DOS**-mogelijkheden zijn in vergelijking met een kloon beperkt. Wie serieus een gecombineerde MS-DOS- en Amiga-optie overweegt is beter af met een 2000-model.
- Enkele oudere software-pakketten draaiden niet onder **Kickstart 1.2.**

Toepassingen

Voor wie is een Amiga 500 nu eigenlijk een aantrekkelijke machine? In tegenstelling tot de op de zakelijke (MS-DOS)-markt gerichte Amiga 2000 valt de 500 in vrijwel ieder kleinschalig computerstraatje. Op de volgende gebieden zou deze 68000-machine zich ons inziens in Nederland heel verdienstelijk kunnen maken, **mits** voldoende soft- en hardware ondersteuning van de grond komt:
* **Onderwijs**. De Commodore 64 is door zijn prijstelling en softwaremogelijkheden redelijk succesvol als leercomputer. Een Amiga 500 kost nauwelijks meer en heeft veel in zijn mars. Vooral op het gebied van interactieve videotoepassingen.
* **Desktop Publishing**. Inmiddels zijn al een aantal printshop- en professionele DP-pakketten leverbaar. Het printerkabel-probleem is nu geëlimineerd en ook laser-Fonts komen binnenkort beschikbaar. De full colour HIRES-graphics van de Amiga 500 maken de paginaopmaak tot een waar genoegen.
* **Business graphics**. Geanimeerde zakelijke grafische presentaties en reklames zijn een ideaal toepassingsgebied voor de Amiga. Er is voldoende software en de prijs maakt de 500 aantrekkelijker dan de investering in EGA-kaarten, animatie-boards en HIRES PC-monitoren.
* **Home Video**. Zie het voorafgaande.
* **Digitale thuismuziek**. Via het MIDI-interface is een compleet kamerorkest mogelijk.
* De **spelcomputer** voor de komende jaren. Spelen als Defender of the Crown en SDI zijn inmiddels zeer geliefd.
* **Experimentele computer** voor de veeleisende hobbyist/programmeur of jonge onderzoeker.
* **Artistieke computer** voor vrije-tijds of professionele kunstbeoefening.
* **Kleine zakencomputer**. De troeven zijn hier: eenvoudige bediening (Intuition userinterface), multitasking, 68000 rekenkracht, telecommunicatie en eventueel de (beperkte) MS-DOS compatibiliteit.

De Amiga 500 biedt veel waar voor zijn geld. Voor rond de *f* 1300,- krijgt de koper een compleet systeem dat talloze hobby-, creatieve en kleine zakelijke wensen kan vervullen. Als voornaamste beperking geldt de voor een groot deel gesloten systeemarchitectuur. Wie het onderste uit de 68000-kan wil hebben loopt bij de Amiga 500 op den duur tegen de geheugen- en uitbreidingsgrenzen aan. Een tweede belangrijk probleem vormt de slappe softwaremarkt in Nederland die, naar wij hopen, spoedig zal aantrekken. ●



Vluchtsimulator op de Amiga 500: grafische kwaliteit is hoog

Zoals gebruikelijk bespreekt Rob Goudriaan weer een aantal klassiekers onder de computerspelletjes. Sommige van deze toppers van toen zijn nog te koop, anderen zult u moeten ruilen of zoeken.

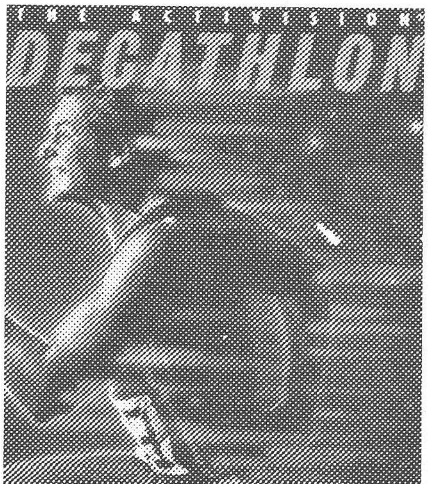
Oud van Goudriaan

Decathlon

Een spel dat enige jaren geleden volop in de belangstelling stond is het aktiespel decathlon. Het spel bestaat uit tien onderdelen, die stuk voor stuk een aanslag plegen op je uithoudingsvermogen.

Om dit spel tot een goed einde te brengen telt niet alleen de lichamelijke konditie maar zeker ook de staat en kwalitëit van het materiaal. Een degelijke joystick is een eerste vereiste. Het is een slopend spel waarbij je na afloop niet alleen het gevoel hebt dat je daadwerkelijk meegedaan hebt, maar ook dat je blij mag zijn als je zonder spierblessures de eindstreep hebt gehaald. Als u uitgerust bent en over een getrainde pols beschikt kunnen we met het spel beginnen, anders is een warming-up misschien beter.

Het eerste onderdeel is de 100 meter hardlopen. Na het startschot be-



weeg je de joystick als een razende heen en weer. Het is zaak om het sneller en regelmatiger te doen dan je tegenstander. Laat je niet afleiden door je tegenstander want tijd kost punten. De volgende onderdelen worden steeds moeilijker en kosten dus ook meer tijd om ze onder de knie te krijgen. En zeg nou zelf, het is toch veel leuker om je tegenstander, bij het polsstok hoogspringen ruglings van zijn stok te zien glijden, dan dat het je zelf gebeurt? Ook het kogelstoten, speerwerpen en niet te vergeten het discuswerpen leveren de meest dolle situaties op, als men ongetraind aan de start verschijnt.

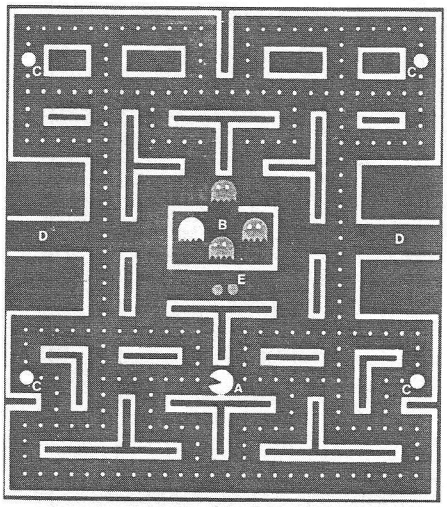
Maar lukt het dan uiteindelijk het projectiel in de lucht te krijgen, dan geeft het een voldaan gevoel het publiek op de tribune te horen juichen. En wat te denken, bij het hoog- en verspringen, van het onderdrukte lachen, als je niet hoger springt dan een gemiddelde kleuter. Dat de tienkamp een zeer zware klus is blijkt wel bij de 400 meter, als je bij de 100 meter dacht het valt nog wel mee, wacht dan maar af. Ga niet te snel van start want zelfs achter de computer hoopt het melkzuur zich op, en krijg je het gevoel op drijfzand te lopen.

Tot slot is er nog de 1500 meter. In dit onderdeel moet alles in de strijd worden geworpen. Bij mij werkt dit onderdeel altijd op mijn lachspieren en vraag niet hoe men met een slappe lach de eindstreep kan halen. Ik heb dan ook het idee dat ik pas over de streep kom als mijn tegenstander allang weer onder de douche uitkomt.

Ik kan er niets aandoen maar ik blijf het een komisch gezicht vinden als twee (volwassen) personen als een razende aan een joystick aan het trekken zijn om vooruit te komen. Kostelijk vermaak ik me dan ook altijd, dit leidt mijn tegenstander echter zo vaak af dat ik dan toch nog win. Decathlon is een spel dat men zeker eens moet spelen. Wie in de computerzaken gaat snuffelen komt het zeker nog wel tegen. Maar men zij gewaarschuwd.

Pac man

Pac man is een spel dat al bekendheid had voor dat de homecomputers hun intrede deden. In elke speelhal waren automaten waar er driftig mee gespeeld werd. Het is een spel wat zeker in een Oud van Goud rubriek thuis hoort. Het is een spel dat veel geimiteerd is, maar nooit geevenaard. Wie kent hem eigenlijk niet, het balletje dat altijd honger heeft en niets anders doet dan blokjes eten en weglopen voor de tegenstanders. Voor degene die het spel nog nooit gespeeld hebben zou ik zeggen: ga naar de winkel, het is vast nog wel verkrijgbaar, en er staan u vele uren plezier te wachten. Na het starten van het spel moet je nog maar een ding voor ogen hebben zoveel mogelijk blokjes opeten. Natuurlijk is dat niet zo gemakkelijk als het lijkt, er zijn een aantal kapers op de kust. Deze lusten dan wel



niet de stippen, maar ze zijn gek op de happelaar, en dan is er weer een kostbaar leven verspeeld. U kunt deze tegenstanders tijdelijk uitschakelen door de gekleurde blokjes, die u in de vier hoeken kunt vinden, op te eten. Uw tegenstanders veranderen van kleur en zijn ineens doodsbang geworden en vliegen voor u weg. Iets onder het midden van het scherm verschijnt er af en toe een vrucht op het scherm. Het verorberen van deze vruchten levert extra punten op. Maar als ik u was ging ik er niet te veel op af, want u bent zo afgeleid en uw tegenstander zit ook niet stil. Het beste kunt u een veilige route kiezen en in de buurt van een stip even op uw tegenstander blijven wachten, dan hap slik en hap tegenstander en u doet de naam happelaar

eer aan. Er zal veel geoefend moeten worden voor er een groot aantal punten op het scherm staat. Dit is een spel dat als de smaak je eenmaal te pakken heeft de nodige nachtrust kan kosten. Eet smakelijk.

The Heist

Heeft u altijd al een grote roofoverval willen plegen, maar daar nooit de tijd of de moed toe gehad, grijp dan hier uw kans. Een geheime organisatie heeft een spion nodig om een chip te pikken die verstopt blijkt te zijn in een schilderij. Maar welke? Zodat er niets anders opzit dan alle schilderijen mee te nemen.

Een galerij vol met schilderijen staat tot uw beschikking. De bewakers zitten met een computer te spelen, zodat de bewaking dus minimaal is (lijkt). Het gebouw bestaat uit drie verdiepingen die verbonden zijn met



een lift. Vanaf een hogere verdieping kan je naar beneden springen. Het lijkt allemaal heel gemakkelijk, maar dat is schijn. Er zijn hier en daar vallen en obstakels opgesteld. Deuren kunnen alleen met een sleutel opengemaakt worden. Er zijn ook bewegende bolletjes, kijk hiervoor uit, ze zijn bij aanraking dodelijk. Als je een sleutel bij je hebt valt de aanraking wel mee dan kost het je alleen de sleutel. Je kan wel over ze heen springen. Hier en daar staan een soort emmertjes met een gekleurd bolletje erin, hiermee in aanraking komen betekent ten alle tijde de dood. Sleutels zijn er overal verstopt. Het is een hele kunst de juiste strategie te bepalen hoe je het beste kunt lopen en welke deur er moet worden opengemaakt. Soms is het beter om naar beneden te springen om zo achter een deur te komen, en een sleutel uit te sparen, voor echte noodgevallen. Met de roltrap kan je snel gaan, maar let erop dat je niet via deze zelfde weg weer terug kunt. In de eerste ronde zijn de lopende banden niet gevaarlijk, maar daarna wordt het anders. Er zijn dan halverwege een soort bewegende balken aangebracht, het is een aparte techniek om het er hier levend vanaf te brengen. Echt in tijdnood kom je met dit spel niet gauw. Je start het spel met twee minuten, en voor elk schilderij dat je pakt wordt de tijd weer op twee minuten terug gezet. Alleen aan het eind als er veel ruimte tussen de schilderijen is, en je wat meer om moet lopen kan het wat krapper worden. Je begint het spel met drie levens en na de eerste 10.000 punten komt er een leven bij. Naar dit spel zult u mischien in de winkels wat meer moeten zoeken, maar het is zeker wel de moeite waard.

Mr. Robot

Mister Robot is een spel dat al jaren geleden uitgebracht is maar bij zeer weinig mensen bekend is. Bij ons staat het al enige jaren op de plank.

Het begint allemaal heel eenvoudig. Met het mannetje links in het beeld kan men bepalen of men er geluid bij wil hebben of niet, door eenvoudig naar de muziektoon te springen. De hele opzet is over de balken te lopen waardoor de stippen verdwijnen. Vergeet er geen een, want teruggaan is soms erg moeilijk. Door tegen het knipperende bolletje te springen krijg je een beschermend kringetje om je heen.

Zolang je beschermt bent kun je de spookjes uitschakelen, maar let op: je hebt hiervoor maar beperkt de tijd. Kijk goed hoe je naar de volgende balk moet, soms is het springen, maar een andere keer moet je je gewoon laten vallen. Doe je het fout, dan kost het een van je vier levens. Als alle blokjes weg zijn gaat men naar veld twee. Het geheel wordt wat moeilijker. Het scala wordt hier uitgebreid met roltrappen en lopende banden. Na het starten van een veld is het raadzaam om eerst eens goed te kijken waar je het beste kunt beginnen. De derde ronde is in verhouding erg eenvoudig, in de vierde is het goed uitkijken want ergens te lang blijven staan, en je vliegt de lucht in. In de vijfde ronde krijg je te maken met trampolines. Het vereist een aparte techniek om hierop stil te blijven staan, dit is echter nodig om niet te ver te springen. In de zesde ronde krijgen we met een techniek te maken om snel te verplaatsen. Je gaat op een transporteerplaats staan, het lijken wel scheermesjes, een druk op de vuurknop en je staat op een andere plaats. Even oefenen om te kijken waar je uitkomt. In de volgende velden krijg je er niets nieuws bij. In ronde 10 komen er magneten om de hoek kijken. Het kost heel wat oefening om de magneten niet tegen je, maar voor je te laten werken. Het zal, de nodige moeite opleveren dit spel tot een goed einde te brengen. Maar voor diegene die het spel zo uitspelen is er de extra mogelijk heid om zelf schermen te ontwerpen. Het is dus een spel dat voor iedereen aantrekkelijk is en blijft.

Oil's well

Stel u voor: u bezit een stuk grond, en volgens deskundigen zit daar olie in. Er is echter één groot probleem u moet het er zelf uithalen. Jaloerse buren en nijdige concurrenten zullen alles doen om u dat te beletten.

Ziehier alle ingrediënten van een spel dat al jaren geleden op de markt is gebracht. Elke keer dat u olie (dus de blokjes) opzuigt levert dit de nodige punten op. De buren hebben via hun eigen grond allerlei vreemde wezens in uw gangen gepompt. zodat u met uw apparatuur niet ongehinderd uw gang kunt gaan. De staalvretende kabels doen niets liever dan uw kabels kapot vreten, en dat ze daar erg goed in zijn blijkt wel dat de kleinste aanraking voor hen al het gewenste effect heeft.

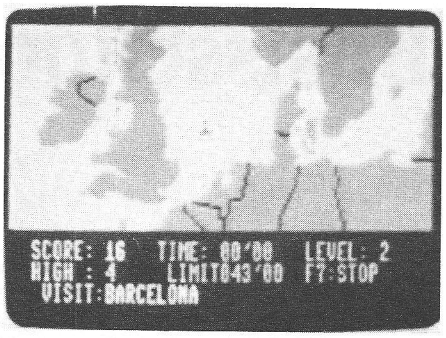
De concurrenten hebben bommen geplaatst die de olieleiding kunnen laten ontploffen. Aan u de taak dit alles te verhinderen en toch zo veel mogelijk punten te verzamelen. Je laat je natuurlijk door niets en niemand tegenhouden en gaat gestaag door de olie op te pompen. Het is mogelijk de staalvretende wezens te vernietigen door ze met de kop van de slang op te zuigen, maar pas op: de slang zelf kan er niet tegen! (en dit kost dan ook een leven). Er dwaalt ook een zandloper rond. Deze doet geen kwaad en levert alleen een heleboel punten op. Veel punten wil zeggen extra levens. Bij elke 10.000 punten wordt het aantal levens met



één verhoogt, en deze zult u later hard nodig hebben. De wat grotere stip die op elk scherm op een andere plaats staat, kan een grote rol spelen. Na deze stip verorberd te hebben gaan alle vreemde wezens veel langzamer, en heeft u de mogelijkheid meer tegenstanders uit te schakelen. Maar pas op, zoals bijna altijd schuilt hier ook een addertje onder het gras: zorg dat je op tijd terug bent in de uitgangspositie, want alles gaat ineens weer over op een razend snel tempo. Goed op de tijd letten, want net als in het gewone leven vliegt de tijd voorbij. Als u alle schermen heeft gehad gaat het spel weer van voren af aan beginnen, alleen veel sneller.

Topografie

Radarsoft, ons eerste Nederlandse software huis heeft een groot aantal Nederlandstallige programma's op de markt gebracht. Waaronder ook enkele goede educatieve spelen. Het topografie-spel Nederland is net als de topografie Wereld en topografie Europa een aantrekkelijk en leerzaam spel.
Na het opstarten van het programma heeft u de keus uit een aantal onderwerpen. Bij de keus *opzoeken* kan er een naam worden ingetypt die dan door de computer op het scherm wordt getoond. Zo kan je plaatsen opzoeken die je zelf niet wist te vinden. Een prima gebruik van de computer als atlas. Bij de keuze *overhoring* moet je laten zien wat je kunt: antwoord geven op de vragen van de computer. Welke rivier is dit, of welke provincie is dit, enz. Grote, maar ook de kleine plaatsen, komen aan bod. Er zijn vier niveau's en vooral de laatste is erg moeilijk. Tot slot is er het *stedenspel*, dat is waarschijnlijk het leukste onderdeel, en niet alleen voor de kinderen. De bedoeling is dat je een helicopter bestuurt en naar de plaats vliegt die de computer aangeeft. Als je vlak bij de plaats bent gaat deze knipperen. In de eerste levels zijn het de grotere plaatsen maar hoe kleiner de plaatsen worden des te moeilijker wordt het spel. Het blijkt dat iedereen beter thuis is met de plaatsen dan met b.v. de rivieren.



Topografie van Nederland, Europa en ook de Wereld, is een leuk en leerzaam spel. Zowel voor de ouderen als ook voor de jeugdigen. Want zeg nu zelf: die saaie rijtjes plaatsen uit je hoofd leren vroeger, was ook niet alles! ●

Een van de grote Nederlandse uitgeverijen op het gebied van educatieve software, Malmberg, heeft een aantal software paketten voor de lagere schooljeugd op de markt gebracht. We bekeken een aantal van deze paketten. Wat als eerste opvalt is de goede verpakking, een stevige doos waarin diskettes en cassetes goed zijn opgeborgen. Elk pakket is voorzien van een duidelijke en in goed Nederlands geschreven handleiding.

Basisschool achter de computer

Malmberg verzamelde software voor de basisschool, geschreven door leraren en opvoedkundigen uit de hele wereld. Het zijn stuk voor stuk programma's waar de nodige aandacht aan is besteed, wat tot uitdrukking komt in de algehele indruk. Het zijn pakketten die in een software bibliotheek op de scholen niet mistaan. Eén van de grote voordelen van deze pakketten is dat we de kinderen zelf hun gang kunnen laten gaan, ze alles zelf laten ontdekken. Dit is de beste manier om ze met een computer om te laten gaan en om ze er vertrouwd mee te maken.

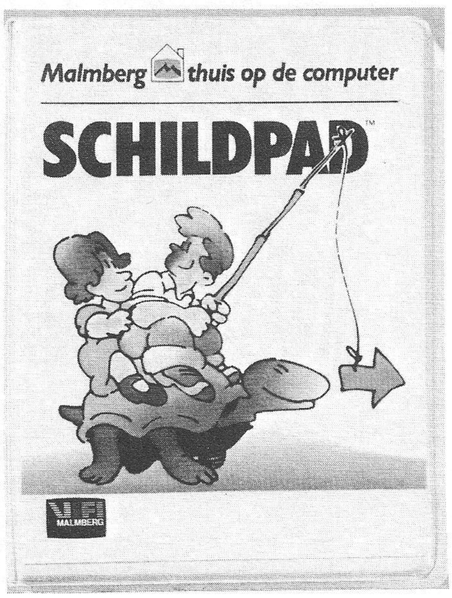
Geheim archief

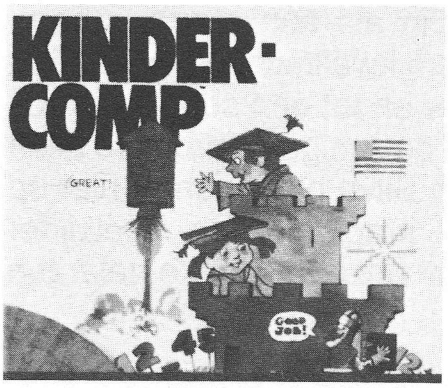
Dit is een programma voor kinderen in de leeftijd variërend van acht tot twaalf jaar. Alle zaken die ze van belang vinden, of dat nu verjaardagen, platen verzamelingen of rapportcijfers zijn, kunnen worden opgeslagen. De naam verraad het al, de bestanden kunnen een naam krijgen die voor anderen geheim is (en blijft). Het geheel is eigenlijk niets anders dan een archief, een verzameling kaartjes met gegevens. Op elke kaart kunnen de kinderen gegevens zetten. Na het opstarten van het programma komt er een menu op het scherm, met de verschilende keuzemogelijkheden. Bij het gebruik van een commodore printer is het geen problemen om de gegevens van de kaarten op papier te zetten. De kinderen kunnen zelfstandig werken met dit programma en het nodigt uit tot experimenteren met de computer.



Schildpad

Met behulp van het programma schildpad is het mogelijk om eenvoudige programma's op de computer te maken, in een speciale en eenvoudige taal. Het is gemaakt voor kinderen van acht jaar en ouder. De taal vertoont een grote overeenkomst met de bekende computertaal Logo. Net als in Logo bestuur je in het programma SCHILDPAD een kleine schildpad. Het programma is opgesplitst in tien lessen. Het begint erg eenvoudig, -praten- met de schildpad. Het is uiteindelijk de bedoeling de schildpad over het scherm te laten lopen. De kinderen leren met dit programma dat er logische en duidelijke opdrachten nodig zijn om het doel te bereiken. Enkele zaken die aan bod komen zijn: werken met kleuren, geluid, zelfs variabelen worden niet vergeten. Les 10 heeft van de maker de naam 'hogere schildpadkunde' meegekregen. Het werken met Schildpad is een duidelijke voorbereiding op latere programeer talen. bijvoorbeeld Basic, waarbij op dezelfde manier moet worden gedacht.





Kindercomp

Kindercomp is, zoals de naam al doet vermoeden, een spelletje voor de kleinere kinderen. Het is de verzameling van een zestal eenvoudige spelletjes. Met wat je van edukatieve software kan verwachten een leerzaam element in elk spel. Het is geschikt voor kinderen vanaf vier jaar. Men kan kiezen uit: *tekenen*, *kladblok*, *toverwoord*, *1-2-3-?*, *a-b-c*, en *zoek dezelfde*. Met deze spelletjes hebben de kinderen een scala van mogelijkheden op de computer. Tekenen, beweging en geluid, het is allemaal geen probleem. Kindercomp is een programma om de lees- en telvaardigheid te ontwikkelen. Als tweede voordeel kunnen we noemen het vertrouwd raken met het toetsenbord.

Speurneus

Het programma speurneus is bedoeld voor de kinderen in de laatste groepen van de basisschool, dus vanaf een jaar of tien. De grote vraag in dit spel is: Spookt het in Mosseldam of niet. Speurneus is aangesteld om dit raadsel op te lossen. Er zijn een aantal verdachten. Zo is er Bert Pioen, de tuinman, een lief oud dametje (mevrouw van Laer), een boze buurman (Alfred Brombeer), een makelaar, een kapitein, een beheerder van een museum maar ook Pietje Peuter. Wie o wie is de dader. Gelukkig heeft Mosseldam ook een computer, waar je gebruik van mag maken. Verder zijn er een aantal handige hulpmiddelen zoals een fototoestel, een speurhorloge, een zaklantaarn en een fototoestel. Alle gegevens over de verdachte personen staan in de computer. Tussentijds kun je het programma stoppen en de gegevens wegschrijven, waarna dan op een later tijdstip het avontuur weer te vervolgen is. Met dit spel



kunnen de kinderen leren aantekeningen te maken, gegevens te verzamelen en deze later weer geordend te gebruiken.



Heeft Pioen het gedaan?





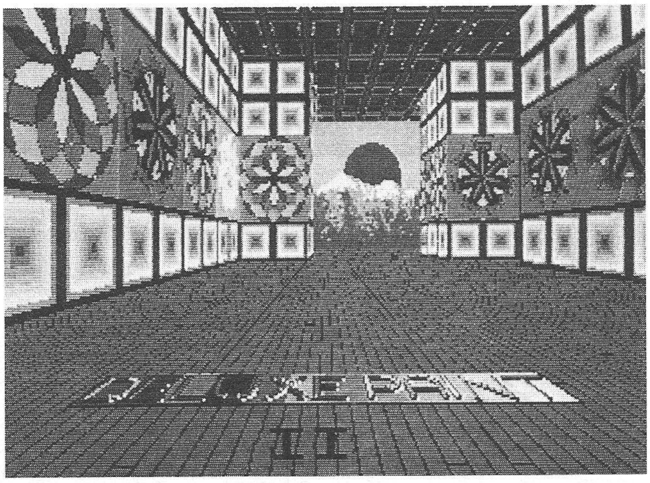
Koppelkaarten

Met koppelkaarten zijn spelletjes te spelen waarbij het aankomt op het trainen van het geheugen en het gebruik van de fantasie. Er kan gekozen worden uit vier onderdelen: *Omgekeerd*, *anagrammen*, *Frans-Engels* en *Hoofdsteden*. Bij het programma *Omgekeerd* moet er een keuze worden gemaakt uit zestien kaarten, waarbij de bijelkaar behorende eruit moeten worden gehaald. Er kan tegen de computer worden gespeeld maar ook een partij tegen een vriendje of vriendinnetje behoort tot de mogelijkheden. De computer houdt de stand bij, er is dus altijd een eerlijke en onpartijdige scheidsrechter aanwezig. Degene die de meeste koppels bij elkaar krijgt wint. Bij het onderdeel anagrammen staan de letters van een woord door elkaar gehutseld op het scherm. Probeer het goede woord te vinden. Met goed opletten is er een heel eind te komen. Bij Frans-Engels is een beetje bekendheid met de talen wel belangrijk. Maar je staat er van te kijken wat kinderen er toch van begrijpen als ze er een tijdje mee spelen. Het onderdeel hoofdsteden is het makkelijkste onderdeel van dit pakket. Zeker voor die leerlingen die op school tijdens de aardrijkskundelessen goed hebben opgelet. Het pakket is gemaakt voor kinderen in de leeftijdsgroep van acht tot twaalf jaar.

DeLuxe Paint sloeg bij de introductie in november 1985 in als een graphics-bom. Nooit te voren waren zoveel grafische mogelijkheden en kwaliteit in een pakket verenigd. Binnen korte tijd werd Deluxe Paint een bestseller en tot een soort graphics handelsmerk van de Amiga 1000. Meer meer dan 50% van de Amiga-gebruikers bezit volgens opiniepeilingen DP. Electronic Arts ontwerper Dan Silva bleef echter niet op zijn lauweren rusten en ontwierp de volledig nieuwe, krachtiger en veelzijder Deluxe Paint II versie. Een reden voor de redactie om onze artistieke talenten eens op het monitorscherm uit te proberen.

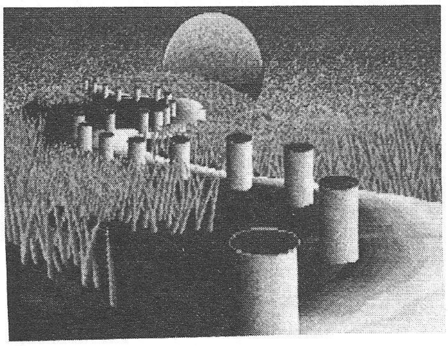
Het onderste uit de graphics-kan?

DeLuxe Paint II



Electronic Arts was een van de eerste softwarehuizen die volledig op de grafische en muzikale mogelijkheden van de Amiga inspeelde. E.A.-software is geen remake van een oud Commodore- of Mac- pakket, maar een volledig voor de Amiga ontwikkeld produkt. Voor het maken van goede HIRES-displays had E.A. dringend de behoefte aan een graphics paint tool. Dat werd Dan Silva's Prism, in feite de oer-DeLuxe Paint. Bij de Amerikaanse doorbraak van de Amiga werd dit stuk grafisch softwaregereedschap geheel voor de MC 68000 aangepast en onder de naam DeLuxe Paint versie 1.0 op de markt gebracht.

Dan Silva is iemand die niet gauw tevreden is. Nauwelijks was versie 1.0 uit of Dan had al tal van nieuwe ideeën en verbeteringen voor bestaande opties in gedachte. Daar kwam natuurlijk nog bij dat de concurrentie, met name Aegis, niet stil zat, en deze zich lekker aan de gestelde DeLuxe Paint-standaard kon optrekken.
Na een jaar van hard werken presenteerde Silva in december 1986



DeLuxe Paint II op de Amerikaanse markt. Volgens Dan bevat versie 2.0 alle opties die bij versie 1.0 nog node gemist werden en loopt het pakket aanzienlijk sneller en vloeiender dan de eerste release. Opscheppen kan uiteraard iedereen, maar zelfs concurrent Aegis moest, volgens perberichten, toegeven dat Electronics Arts opnieuw een imposante grafische Amiga-standaard gecreëerd had.
DeLuxe Paint II is dus geen upgrade, maar een compleet nieuwe, zij het volledig IFF-compatibele, DeLuxe Paint-uitvoering.

Wat is nieuw?

Op het eerste gezicht lijkt DeLuxe Paint II nog veel op versie 1.0. Wie echter wat verder kijkt dan zijn/haar artistieke neus lang is ontdekt al spoedig een hele set nieuwe krachtige opties en verbeteringen van oude manco's.
Nieuw zijn de opties:

- **Stencil** vervult een soort maskerfunctie die bepaalde schermdelen afdekt tegen ongewenste veranderingen.
- **Fixed Background** maakt het mogelijk om voorstellingen en kleurlagen tegen een bestaande achtergrond aan te brengen zonder dat deze aangetast wordt.
- **Verschillende screenformats**. Low-, medium en high resolution, interlace, overscan (geeft een compleet, dat wil zeggen schermgroot, videobeeld) en een tekenvel groter dan het monitorscherm.
- **3D-perspectief**.
- **contour-bijwerk mogelijkheden** zoals Smooth en Anti-Alias.

- Diverse nieuwe **Fills**.
- **Multi Color Cycle**, de meerkleurige uitvoering van color cycling.
- Uitgebreidere **zoomopties**.
- **Multitasking** middels het booten vanuit de Workbench.

Alsof dit nog niet voldoende is zijn er minder in het oog springende verbeteringen aangebracht, die echter minstens even waardevol zijn:

° Een compleet nieuw 150 pagina's tellend handboek met gebruikerslessen.

° Een aanzienlijk snellere werking van de verschillende programmaonderdelen. Bijvoorbeeld het zonder opnieuw te hoeven booten wisselen tussen de verschillende schermresoluties en kleurschakeringen.

° Het optimaal gebruik van RAM-uitbreidingen via het supergrote tekencanvas van 1008 x 1024 pixels.

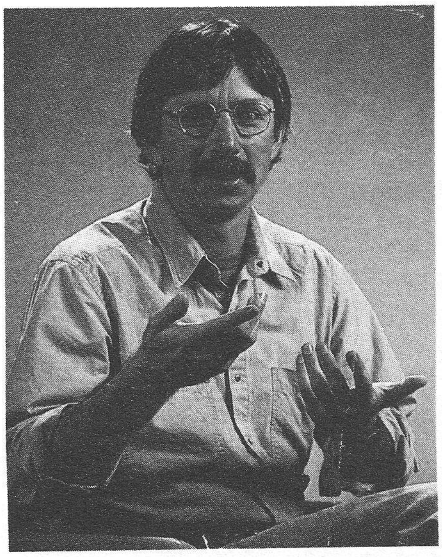
° Een onbeperkte keuze uit tekenpenselen.

° Vrije toegang tot alle tekststijlen en fonts.

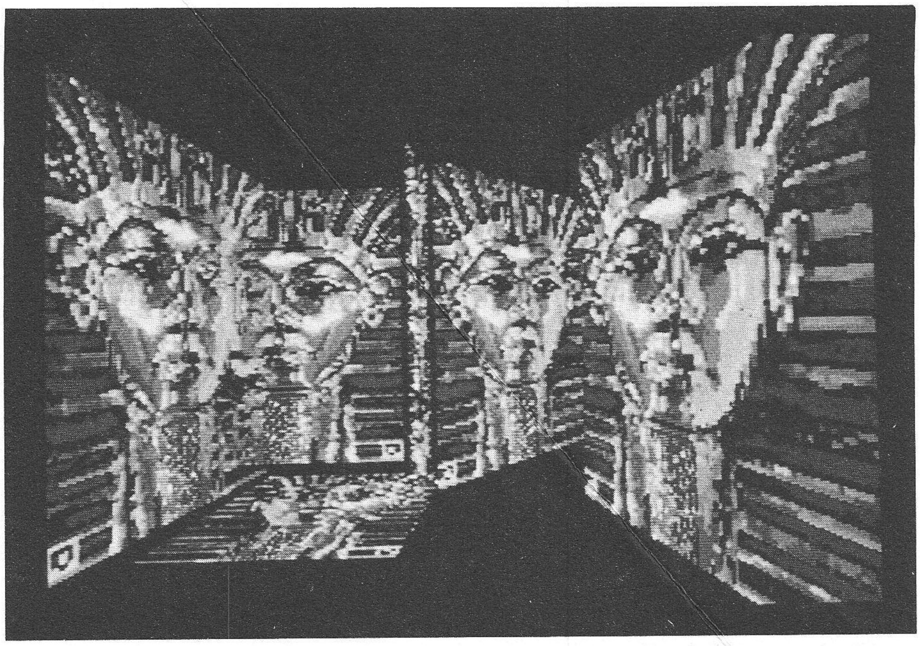
° De mogelijkheid om een perseel in elke gewenste vorm te modelleren.

° Een druk op de HELP-toets is voldoende om bij het zwart slaan van het monitorscherm (ten gevolge van een verkeerde kleurenkeuze is er niets meer te zien) weer naar de vorige situatie terug te keren.

Hoe langer je met DeLuxe Paint II werkt des te meer mogelijkheden je



Dan Silva



ontdekt. Wij moeten ons hier helaas beperken tot een korte bespreking van de krachtigste opties.

Stencil en Fixed Background

Bij het ontwerpen van de **Stencil-optie** heeft Dan Silva goed naar de techniek van het airbrushen gekeken. Om te voorkomen dat delen van de tekening per ongeluk meegespoten worden dekt de artiest hen met een transparant masker af.
Ook bij DeLuxe Paint II kan de gebruiker zijn/haar tekening met een doorzichtig Stencil afschermen en zo nauwkeurig het paintproces controleren. Er kan zowel voor als achter het gestencilde object geschilderd worden.
Het gebruik van **Fixed Background** lijkt veel op de toepassing van acetaat overlays. Een tekening wordt als het ware in de achtergrond gefixeerd en kan zo zonder schade aan het origineel overgeschilderd, afgekrapt en zelfs in delen verplaatst worden.
Beide opties besparen de artiest veel overmaak-werk en tijd.

Fills

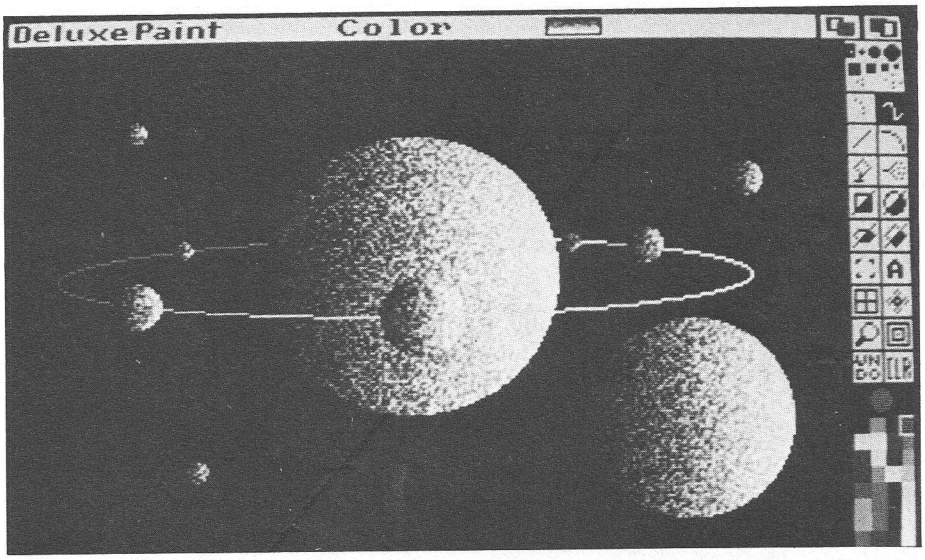
DeLuxe Paint II kent een aantal verbeterde en uitgebreide Fill- opties.
Pattern Fill maakt het mogelijk om met de penseel elk gewenst patroon of kleur in elke grootte, vlakken op te vullen. Alle zelf getekende voorwerpen kunnen ook weer als een penseel of Fill-patroon ingezet worden.
Perspective Fill is een krachtig grafisch tool om 3D- kleurgradienten te creëren.
Gradient Fill biedt horizontale en vertikale graduele Fills in 32 kleuren. De mate van kleurscheiding hangt daarbij af van de zelf in te stellen Dither-waarde. 0 is keihard. Maximum geeft een superzachte overgang.

De screens

Een probleem voor de videohobbyist was het feit dat een DeLuxe Paint-scherm net iets kleiner bleek dan de beeldvullende video- display. DeLuxe Paint II biedt de desktop videoer nu een **Full Video** randloos kleurenbeeld. Met behulp van een *Genlocker* plakt u moeiteloos een fantasie-achtergrond bij een normaal camerabeeld of computersprites voor een real-life scenery.

Het werken met DeLuxe Paint II geeft het gevoel te componeren met kleuren en beelden.

De maximum grootte van het tekencanvas, het **virtual screen**, bedraagt bij versie 2.0 1008 x 1024 pixels. Bij voldoende RAM- geheugen kunnen zo gemakkelijk meerdere planes van meer dan 1 miljoen pixels met elkaar gecombineerd worden. Ook is het met 1MB of meer RAM mogelijk om meerdere schermen in de actieve mode te hebben. Alleen de hardware is nog maar beperkend in tegenstelling tot de oude situatie waarbij de grafische software de broekriem aantrok.
Een dankbare verbetering is het binnen DeLuxePaint switchen tussen de



verschillende grafische modes. Bij versie 1.0 was daarvoor een reboot nodig. Bij versie 2.0 voldoet een simpel muis-commando en de schermtekening krijgt een andere res-mode tijdens het painten.
Tot slot noemen wij nog de interlace-modus waarbij de videoliefhebber 32 kleuren in 320 x 400 mediumres kan onderbrengen. Uiteraard neemt het flikkeren van het beeld op 25Hz (i.p.v 50Hz) wel iets toe.

2D/3D

Door middel van **Perspective Drawing** kan de tekenaar met een muisklik tussen 2D- en 3D-perspectief switchen. In de 3D-modus kan de verfkwast zowel in de X-, Y- als Z-richting bewogen worden. Er zijn twee verschillen 3D-modes: **Perspective Mapping** en **Perspective Fill**. De penseel past zich qua grootte automatisch aan de perspectivistische vertekening aan. Dat hadden onze beroemde landschapsschilders uit de gouden eeuw eens moeten zien....
Perspective Mapping tekent met de penseel een 3D-matrix (grid) die later met behulp van Perspective Fill gradueel wordt opgevuld. De kleurgradient creëert een extra 3D-dimensie.

Tekenhulpjes

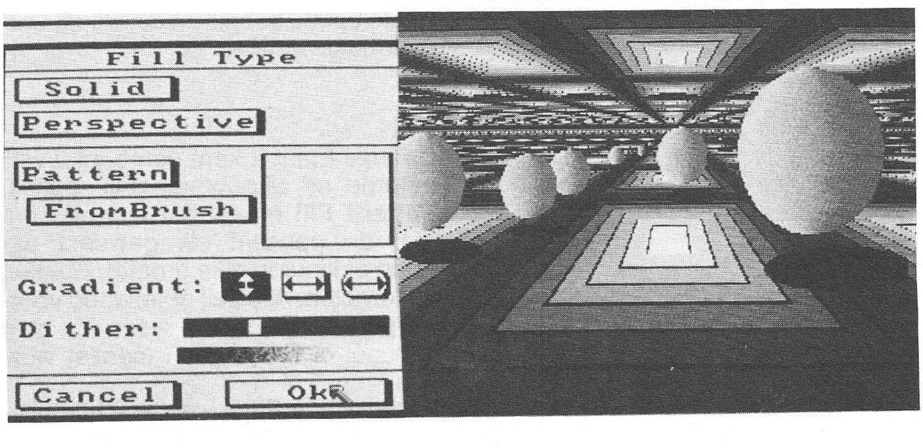
De tekentools van DeLuxe Paint II vallen in de volgende kategorieën uiteen:

- **Color en resolution** tools. Er zijn zes kleuren-controls voor het besturen van het mengpercentage aan rood, groen en blauw, de kleurverzadiging, het contrast en het kleurzweem. De details en shades kunnen ook achteraf nog gewijzigd worden. Kleurovergangen worden automatisch tussen de twee opgegeven overgangskleuren gegenereerd. Er zijn 16 (in interlace 32) kleuren uit een pallet van 4096 mengkleuren beschikbaar.
- De **Penselen**. Bij DeLuxe Paint II kan werkelijk alles als tekenpen of verfkwast gebruikt worden. Er zijn standaard brushes (cirkels, rechthoeken, punten), zelf te ontwerpen brushes en de **corral**-functie om een onderdeel van de tekening als brush in te zetten.
- De **Zoomoptie** biedt de artiest een uitvergroting van een deel van de tekening waarop alle paint-tools gebruikt kunnen worden. Met de scroll-optie kan de tekenaar de hele plaat op zijn gemak doorlopen. Via spilt-screen zijn gelijktijdig het uitvergrote deel en de normale tekening te zien.
- De **bijwerkopties**. Het tool **Smooth** vult de contouren rond gedigitaliseerde beelden op. Donkere banden rond dergelijke objecten behoren daarmee tot het verleden. **Anti-Alias** werkt vervormde en gerafelde uiteinden van figuren bij voor het maken van vloeiende overgangen.
- **Symmetrish tekenen** via spiegelen, dakpansgewijs of cyclish herhalen van figuren of objecten.
- **Grids** voor het precies op schaal tekenen.
- **Multi Cycle** creërt een veel kleurige waterval van tinten. Deze optie is heel geschikt voor het maken van animatie-effecten.

Voor wie helemaal niet kan tekenen is er nog altijd de mogelijkheid om beelden te digitaliseren en vervolgens artistiek te bewerken. Bijvoorbeeld met het geheel DeLuxe Paint compatibele Digi-View van NewTek zijn via de Hold and Modify-mode adembenemende effecten te bereiken.

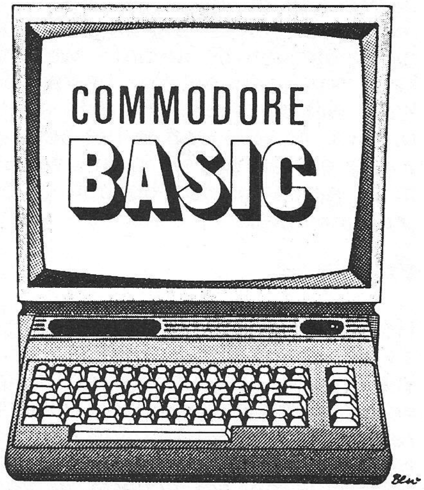
DeLuxe Paint II slaat een krachtige brug tussen de Amiga, artiest of desktop/video-werker. De geboden mogelijkheden slaan tot nu toe alles wat op een gewoon PC-tje mogelijk was. Er blijft natuurlijk altijd wat te wensen en verbeteren over, maar wie meer wil zal ook vele malen zoveel voor hard- en software moeten neertellen. DeLuxe Paint II zal in Nederland tussen de ƒ350,- en ƒ400,- gulden gaan kosten. Of de Amerikaanse upgrade-kit voor versie 1.0 ook in ons land beschikbaar komt, was bij het ter perse gaan van dit nummer nog niet bekend.

●



We zijn inmiddels al een eind op weg met onze Basic cursus. In de afgelopen 13 lessen hebben we al heel wat over Basic kunnen leren en de nodige programma's geschreven. De moeilijkheidsgraad is daardoor inmiddels gestegen tot het niveau van een aankomend programmeur. Toch probeert Jan Bodzinga, waar het maar mogelijk is, de stof voor iedereen begrijpelijk te houden. Daardoor kunnen ook de nieuwe lezers van dit blad nog altijd 'aanloggen' en meedoen. De cursus is zo opgezet, dat zelfs de nieuwelingen op computergebied stap voor stap kunnen worden ingewijd in de geheimen die de computer voor ons in petto heeft.

Basis Basic



Deel 14: Een Database (III)

In de voorgaande afleveringen zijn we begonnen met de opzet voor een professioneel ogend database-programma. De listing is opgezet voor een adressenbestand, maar kan door de gebruiker zelf gemakkelijk worden aangepast aan andere door hem of haar gewenste toepassingen. Het geheel is een vrij ingewikkeld Basicprogramma, maar door een goede structuur.en opzet van de listing is het relatief eenvoudig om de diverse subroutines te kunnen lokaliseren en eventueel aan te passen.

Tot op dit moment hebben we alle universele subroutines die in het programma voorkomen behandeld, terwijl ook het hoofdmenu en het toevoegen van gegevens als routines inmiddels zijn behandeld. Vooral de principes van gegevensbestanden komen aan de orde. Aan het eind van deze les zijn we zeker weer een heel stuk verder met dit fraaie stuk programmeerwerk.

Wat vooraf ging.

De vorige afleveringen hebben we voor het grootste deel gewijd aan het opzetten van dit programma. Als eerste zijn we uitvoering ingegaan op de diverse **arrays**. Een array is een verzameling van gegevens, die onder dezelfde noemer in een programma voorkomen. We kunnen op die manier erg veel, min of meer identieke, gegevens gemakkelijk verwerken. Daardoor zijn we zo flexibel, dat het voor een programma weinig uitmaakt, of we nu bijvoorbeeld 20 **records** of 2000 records in het programma willen verwerken. Uiteraard is het beschikbare geheugen daarbij wel van belang. Het programma in casu verandert echter in 't geheel niet door de grote hoeveelheid data.

De opzet van ons programma is structureel. Dat wil zoveel zeggen als voor iedere bewerking of onderdeel daarvan een strict eigen gebied in de programma-listing is gereserveerd, waar dan ook niets anders kan en mag gebeuren. Ofwel als er een invoer voor **JA of Nee** wordt gevraagd, dan hebben we daar een aparte (sub)routine voor. Zo gaat het ook met alle andere bewerkingen in het programma. Onze listing is verdeeld in 13 routines. We zullen voor de volledigheid het geheel nog even op een rij zetten:

***0000 - 0100REM programmatitel**
*0100 - 0300Initialisatie
*0300 - 1000Diverse subroutines
*1000 - 2000Programma (start)menu
*2000 - 3000Toevoegen
3000 - 4000Muteren
4000 - 5000Verwijderen
5000 - 6000Printen
6000 - 7000Sorteren
7000 - 8000Zoeken
8000 - 9000Inlezen
9000 - 9500Wegschrijven
9500 - 9999Stoppen

Al deze onderdelen zitten op een vaste plek in de complete listing. We hebben dit zo geschreven, om ook later op een snelle en handige manier wijzigingen en verbeteringen in het programma te kunnen aanbrengen. We hoeven maar op het programma-schema te kijken om te weten in welke regels we een bepaalde module moeten zoeken. Daardoor lopen we de minste kans om op een verkeerde plaats de wijzigingen in het programma aan te brengen, wat op z'n beurt weer een grote tijdwinst kan betekenen. De met een (*) asterix gemerkte modules zijn in de vorige lessen uitvoerig aan de orde gekomen. Aan het eind van deze les zullen we een complete listing geven van alle tot nu toe behandelde onderdelen van het programma.

Muteren.

Het muteren of wijzigen van gegevens in ons adressenprogramma is een wezenlijk onderdeel van goed gegevensbeheer. Er zijn in het verleden al heel wat theorieën over gemaakt, hoe dat optimaal in z'n werk moet gaan. We zullen ons met deze ingewikkelde materie niet verder inlaten dan nodig is. Wel moet je weten, dat het wijzigen van een record inhoudt, dat oude data die in een be-

paalde record aanwezig zijn worden vervangen door nieuwe gegevens.
Het muteren kan bijvoorbeeld nodig zijn, als één van de personen die in ons adressenbestand voorkomt een ander adres heeft gekregen. Ook bij **aanvulling** van de bestaande gegevens moet er worden gemuteerd. De grootte van het record in de array blijft bij het muteren echter altijd gelijk. Alleen de **data** in het record kan wijzigen. We moeten ons eerst terdege afvragen wat er allemaal wel en juist niet gebeurt, als we een record geheel of gedeeltelijk gaan wijzigen. Als eerste (en enige) worden de gegevens in het beweuste record veranderd. Dat betekent, dat de lengte van dit record - we werken met sequentiele bestanden- kan wijzigen. Het totaal aantal records, of adressen in het bestand wijzigt echter niet. Dit is iets, wat we in de gaten moeten houden bij het programmeren.

Principe

Hoe gaat nu het wijzigen op papier in z'n werk ?
We moeten natuurlijk eerst weten welk record we willen gaan veranderen. De meest omslachtige methode is, om alle bestaande records stuk voor stuk op het scherm te laten verschijnen en bij elk adres te vragen of dit wellicht veranderd moet worden. Dat is uiteraard een leuk tijdverdrijf, maar handig is het allerminst. We zullen er iets op moeten vinden, om tamelijk snel het te wijzigen record boven water te krijgen. De eerste bewerking bij het wijzigen bestaat dus uit het **opzoeken** van het bewuste record. Het gezochte record moet ter controle op het scherm worden **geprint**. Daarna moeten we via communicatie met de gebruiker vasstellen of dit het **juiste adres** is wat moet worden gemuteerd. De derde stap is dus het definitieve besluit om het zojuist gezochte adres te gaan wijzigen.
Als vierde in de reeks handelingen moeten we de nieuwe gegevens per veld **invoeren**. Daarna volgt opnieuw een **controle** op de ingevoerde gegevens en als laatste moeten de oude gegevens uit het bestand (de tweedimensionele array) worden **vervangen** door de nieuwe, zojuist ingetypte data.
Al met al een hele rij handelingen, die we stuk voor stuk moeten afwerken tijdens het muteren:

MUTATIE-batch
1Zoeken
2Printen
3Controle
4Invoeren
5Controle
6Vervangen

De eerste bewerking, het zoeken, vinden we ook terug als een aparte module in het hoofdmenu. De daarvoor gereserveerde programmaregels bevinden zich tussen 7000 en 8000. Het zou dom zijn, als we voor het werken met het mutatie-gedeelte geen gebruik van de nog te schrijven module voor het zoeken zouden maken. Daarom lijkt het verstandig eerst de zoekmodule te schrijven en te bespreken.

Zoeken

Het opzoeken van records gaat in ons programma vanuit het computergeheugen. We zijn in deze serie nog niet zover, dat we een adressenbestand vanaf de tape of disk zullen gaan benaderen. Hier lopen we ook tegen één van de grenzen van de mogelijkheden van het programma op. Het maximum aantal adressen wat in ons bestand kan worden verwerkt is afhankelijk van de grootte per record en uiteraard de ruimte aan vrije bytes in het computergeheugen. Gemiddeld neemt een record zo'n 100 bytes geheugen in beslag, waardoor we dus met **20.000 Bytes Free** nog altijd 200 adressen per bestand kunnen verwerken. Daarnaast is het natuurlijk altijd mogelijk om meerdere bestanden met verschillende namen met dit programma te kunnen verwerken. Het opzoeken van records in dit adressenbestand kan vanwege de (geschatte) maximale hoeveelheid adressen per bestand gebeuren op de meest simpele manier. Dat betekent, dat de gebruiker wordt gevraagd, op welke veldsoort (naam, straat, postcode etc.) er moet worden gezocht. Nadat het veldtype door de gebruiker is ingegeven, gaan we met een **lus** door het complete bestand, om de gezochte code te vergelijken met de inhoud van het bewuste veld. Vinden we een '**match**', waar zoekcode en veld gelijk zijn, dan wordt het complete record op het scherm geprint. Daarna zoeken we verder naar een eventueel volgend identiek veld, om ook dit record te printen.... en zo verder tot het eind van het bestand.
Het zoeken gaat in de echte databases meestal op een wat geavanceerder manier. Men gaat dan uit van een zogenaamde 'zoekindex'. Dat betekent, dat het hele bestand moet worden gecatalogiseerd; er wordt een aparte '**index**' van gemaakt, waarin dan alleen van bijvoorbeeld het veld 'Postcode' de gegevens voorkomen, met daarachter het (logische) nummer van het record. Op die manier is er toch nog heel snel te zoeken en te sorteren op enorme hoeveelheden adressen. In ons geval, met vaak niet meer dan 200 entries, hoeven we niet zo ver te gaan, dat we aparte indexen bouwen voor de velden waarop het meeste naar gegevens zal woren gezocht.

Zoekmethode

Nadeel bij de door ons gevolgde methode is natuurlijk de snelheid. Daar zullen we echter niet zoveel van merken, omdat de bestanden relatief klein blijven. Het grote voordeel van onze zoek-module is echter het feit, dat we zo hebben geprogrammeerd, dat het voor de gebruiker mogelijk is, om op ieder gewenst veld het bestand af te 'scannen' op zoek naar de gevraagde gegevens. En dat alles ook nog, zonder veel extra regels aan het programma toe te voegen. Het geheim van deze manier schuilt in de **twee-dimensionele array** A$(.), die we gebruiken om de adresrecords in op te slaan. Laten we de listing van het zoekgedeelte maar eens bekijken:

```
7000GOSUB 500 : REM kopprint
7010PRINT:PRINT"Geef sleutelveld
:"
7020FOR II = 0 TO 5
7030PRINT II+1;" - ";VN$(II)
7040NEXT II
7050PRINT : PRINT "Maak een
keuze ";
7060GOSUB 800 : REM lees toets
7070KK= VAL(T$)
7080IF KK OR KK 7 THEN 7050
7090IF KK = 7 THEN
RETURN:REM menu
7100VL=KK-1 : REM veldsoort
```

Het eerste deel van de zoekmodule houdt zich bezig met de invoer van het '**sleutelveld**'. Daarmee wordt het veld in het adressenrecord bedoeld, waarin het **zoekgegeven** moet worden gezocht. Om deze routine zo universeel en efficiënt mogelijk te maken, kunnen we -zoals eerder opgemerkt- de zoekcode ingeven voor *alle* beschikbare velden. Weliswaar gaat ons programma niet verder dan slechts één zoekcode per keer, maar het is niet erg moeilijk dit later zelf uit te breiden tot meerdere condities. Dan zou je bijvoorbeeld alle namen die beginnen met een 'B' en daarnaast een postcode hebben die begin met '12' uit het bestand

kunnen '**filteren**'. Ons geval beperkt zich tot één conditie. Maar deze routine kunnen we wel op ieder veld van de array loslaten.
Met hulp van de serie vaste subroutines bepalen we de keuze voor de veldnaam. Alle namen worden geprint op het scherm (regel 7010 - 7090). Daar zien we, dat eerst de kop met de bewerking wordt geprint op het scherm (sub 500), waarna in de lus II alle 6 veldnamen (VN$()) in beeld komen. We gebruiken vervolgens subroutine 800 v.v. om de ingetypte keuze te verwerken. De keuze, variabele KK wordt getest op z'n geldigheid, waarna de veldnaam-variabele (VL) de waarde krijgt van KK-1. Waarom deze waarde 1 lager ligt dan KK moet je zelf maar eens proberen uit te zoeken.
We zijn inmiddels toe aan het volgende gedeelte van de zoekroutine:

```
7110VLAG=0
7120GOSUB 500 : REM kopprint
7130PRINT:PRINT:PRINT
7140PRINT "zoeken op ";
VN$(VL);" :";
7150INPUT Z$
7160FOR I = 1 TO TT : REM loop alle
records
7170IF LEFT$(A$(I,VL),LEN(Z$)) =
Z$ THEN GOSUB 7210:IF SU=99
THEN I = TT
7180NEXT I
```

Dit gedeelte houdt zich bezig met het echte scannen van het gehele bestand. Eerst wordt nogmaals een (nieuwe) kop geprint, waarna in regel 7150 de invoer van de op te zoeken string in variabele Z$ wordt gezet.
In de lus I, die loopt van adres 1 tot adres TT, - het totaalaantal- wordt vervolgens op veld VL in de array A$(I,VL) bekeken of Z$ overeen komt met **hetzelfde** deel in dat veld. Deze test staat in het eerste gedeelte van regel 7170. Deze regel bevat dan ook de kern van de hele zoekroutine. We testen alleen dat gedeelte van de string dat even lang is als Z$. Dit gebeurt door het gebruiken van de functie **LEFT$()**. Op deze manier is het ook mogelijk alle namen die bijvoorbeeld beginnen met een "S" op te zoeken. De lengte van Z$ - LEN(Z$)- is dan natuurlijk 1, waardoor ook van alle te testen namen in A$(I,VL) via LEFT$() alleen het eerste karakter wordt bekeken. Dit gaat niet alleen sneller, maar is ook veel handiger, omdat we op deze manier ook namen en andere gegevens waarbij we niet helemaal zeker zijn van spelling en dergelijke gemakkelijk uit het bestand kunnen filteren. Wordt er inderdaad een goede string gevonden in array A$(I,VL) dan wordt subroutine 7210 v.v. uitgevoerd. Over de variabele SU zullen we het later nog hebben. In ieder geval wordt aan het eind van regel 7170 de lus beëindigd door de waarde van I gelijk te stellen aan TT, als SU=99. Daardoor wordt in regel 7180 niet opnieuw via NEXT teruggesprongen naar het begin van de lus. In alle andere gevallen wordt deze routine afgesloten met een **NEXT** tot I de waarde TT heeft bereikt en het gehele bestand is afgezocht.

Vlaggen

Als laatste opmerking bij deze programmaregels wil ik nog wijzen op de variabele **VLAG** in regel 7110. Deze wordt elke keer dat de zoekroutine wordt aangeroepen op 0 gezet. Aan de hand van de vlagwaarde zullen we verder op bekijken of we al dan niet een 'match' hebben gevonden tussen Z$ en de inhoud van één of meer indexen in ons adressenbestand. Daarvoor hebben we regel 7190:

```
7190IF VLAG = 0 THEN 7380
7200RETURN
```

Als vlag nog steeds de waarde 0 heeft, wordt in regel 7190 een aparte subroutine aangeroepen, die zit op 7380. Deze routine meldt op het scherm, dat er geen enkel veld is ontdekt in het bestand, waarin de gevraagde string **Z$** voorkomt. Het is uiteraard een goede gewoonte om dit soort fouten apart aan de gebruiker door te geven. Vandaar routine 7380 - 7420:

```
7380REM record niet aanwezig
7390PRINT Z$ " niet gevonden"
7400PRINT "Typ toets voor vervolg"
7410GOSUB 800
7420RETURN
```

Veel is er niet te zeggen van deze schermmelding, waarbij keurig wordt gewacht tot de gebruiker een toets heeft ingedrukt om aan te geven, dat hij de melding heeft gelezen.

Match

Veel interessanter dan het voorgaande is het programmagedeelte wat wordt uitgevoerd, als Z$ matcht met een gedeelte van een veld. De aanroep voor deze subroutine vinden we in regel 7170:

```
7210REM record gevonden - print
op scherm
7220GOSUB 500
7230VLAG =1

7240PRINT "Record nummer : ";I
7250FOR II = 0 TO 5
7260PRINT VN$(II)" : "A$(I,II)
7270NEXT II : PRINT
```

Na de kop vinden we hier het nut van variabele **VLAG**. Zodra er een match optreedt, wordt naar deze routine gesprongen en wordt variabele **VLAG** 1. Daardoor is het mogelijk de test op het al of niet vinden van een goede string aan de hand van de waarde van **VLAG** te bepalen. Vervolgens wordt in lus II zowel de veldnamen als de gegevens in array A$() op het scherm geprint. Merk hierbij op, dat de index van het complete gevonden record wordt bewaard in **lusvariabele I**, die we hebben gegenereerd in regel 7160. Het is belangrijk dit te onthouden voor een paar andere modules in ons programma, die ook gebruik maken van deze zoekmodule.

De module-vlag SU

In onze subroutine gaan we verder vanaf regel 7280 met variabele SU. Deze SU heeft in ons geval drie mogelijke waarden: 0, 1 of 2:

```
7280ON SU+1 GOTO 7300, 7340,
7340 : REM SU=SOORT MODULE
7290REM SU=0 normaal zoeken
7300PRINT "Typ toets voor vervolg"
7310GOSUB 800
7320GOTO 7370

7330REM SU=1 Verwijderen SU=2
Muteren
7340PRINT "Dit record bewerken
(J/N)"
7350GOSUB 850
7360IF T$="J" OR T$="j" THEN
SU=99
7370RETURN
```

We kunnen SU ook beschouwen als een soort 'flag', die door zijn waarde aangeeft waar we precies mee bezig zijn. Op die manier kunnen we van eenzelfde subroutine op meerdere wijzen gebruik maken. Heeft SU de waarde 0, dan betekent dit, dat we bezig zijn met een kale zoekactie, die vanuit het hoofdmenu wordt aangestuurd. In regel 1010 vinden we dan ook de opdracht (LET) SU=0. De waarde van SU wordt daar iedere keer dat we bij het hoofdmenu terecht komen op 0 gezet. Als SU een waarde heeft van 1, betekent dit, dat we de zoekroutine aanroepen **vanuit** de routine om een

record te verwijderen. We komen daar straks wel op terug. Ook voor het muteren (wijzigen) van een adresrecord, moeten we eerst zoeken naar het bewuste record in de array. Omdat we slimmer dezelfde routine voor het zoeken kunnen gebruiken, die toch al aanwezig is in het programma, geven we dit aan, door in de mutatie-routine SU de waarde 2 toe te kennen. Tot slot krijgt SU nog een speciale waarde, namelijk 99 (erg symbolisch) als er specifieke dingen met een record gaan gebeuren. De test op SU voeren we uit in regel 7280, met een **ON .. GOTO opdracht**. Omdat SU ook de waarde nul kan hebben, moeten we er eerst 1 bij tellen, om dit commando goed te laten functioneren. Het is uiteraard helemaal niet nodig, om in ons geval met een **ON..GOTO** te werken, maar wel beter voor de structuur en zeker te verkiezen als we nog meer waarden aan de SU vlag willen toekennen.
Als SU=0, dan kunnen we volstaan met een opdracht om een toets in te typen, als het gevonden record, wat op het scherm staat, door de gebruiker is bekeken. De routine gaat via sub 800 vv. door naar de zoeklus, om naar nog meer eventuele gelijke string te zoeken.
Is SU 1 of 2, dan moet er iets anders gebeuren. We moeten nu namelijk weten, of er met het specifiek gevonden record iets moet gebeuren, of dat er verder moet worden gezocht, naar een andere 'match' in het bestand. Aan de hand van een eenvoudige JA/NEE vraag (regel 7340) bepalen we of het gevonden record het juiste adres bevat. Is dit zo (regel 7360) dan geven we SU de waarde 99 om dat aan te geven. Is het tegendeel waar, dan houdt SU z'n oorspronkelijke waarde en keren we ook terug naar de zoekroutine. Heeft SU de waarde 99 dan zien we bij terugkeer in de echte zoekroutine, dat aan de hand van SU=99 de lus en de routine beëindigd wordt, om terug te gaan naar de module die het zoeken heeft geactiveerd.

Subroutines

Het is verstandig deze zoekroutine in z'n geheel eens grondig te bekijken, omdat er nogal wat handigheidjes in zitten verstopt, die we in eerdere lessen al uitvoerig hebben bekeken, maar wellicht in het normale programma-gebruik niet meer zo opvallen. Let maar een op hoe vaak we nu al bepaalde kleine vaste subrouti nes hebben aangeroepen, en met welk doel dit is gebeurd. Je zult verbaasd zijn over de diverse mogelijkheden die verscholen zitten in een paar regels programma. Het is daarbij helaas niet zo, dat je kunt leren programmeren, door steeds maar listings van anderen klakkeloos over te nemen. Dan kun je zelfs beter meteen naar de winkel gaan om er een echt goed pakket voor je toepassingen te kopen. Wil je zelf aan de slag, dan moet je niet alleen een goede analyse uitvoeren op bestaande listings, zoals deze, maar daarnaast ook helemaal zelf aan het schrijven. Ervaring is in dit geval de beste leermeester. Ook van eigen (en andermans) fouten valt een heleboel te leren.

Muteren II

We hebben nu de complete listing van de zoekroutine achter ons. Daar zijn we eigenlijk op gekomen, omdat we ontdekten, dat, voordat er ook maar een record kan worden gewijzigd, dit eerst in het bestand moet worden opgezocht. In de zoekroutine hebben we aan de eerste twee bewerkingen tijdens het muteren voldaan, **zoeken & printen**. We zullen daarom de feitelijke mutatie-routine eens bekijken:

```
2999 REM Muteren / Wijzigen
3000 GOSUB 500: REM kopprint
3005 SU=2: GOSUB 7010 :IF SU = 2
THEN RETURN
```

We zien in de eerste regels, dat SU hier inderdaad de waarde 2 krijgt. Bovendien printen we een eigen kop, met hulp van subroutine 500. Vervolgens gaan we naar de zoekroutine, door het aanroepen van **GOSUB 7010**. Als SU terug komt met een waarde 2, dan betekent dit, dat we tijdens het zoeken **geen** record hebben gevonden, wat we willen wijzigen. Daarom kunnen we zonder problemen terug naar het hoofdmenu met een **RETURN**-opdracht. Heeft SU een andere waarde dan 2, in ons geval dat symbolische 99, dan gaan we verder met het wijzigen van het hele record:

```
3010 GOSUB 500: REM KOPPRINT
3011 FOR II = 0 TO 5
3012 PRINT VN$(II);": ";A$(I,II)
3014 NEXT II
3020 PRINT:PRINT "Wijzigen zeker
(J/N)"
3030 GOSUB 850: IF T$= "N" OR
T$="n" THEN RETURN
```

In bovenstaand gedeelte wordt (opnieuw) het hele record op het scherm gezet, waarbij we ter controle de vraag stellen of dit het juiste record is. Geven we hier een ontkennend antwoord, dan wordt ook terug gesprongen naar het hoofdmenu en gebeurt er niets met de data in het record.
In alle andere gevallen gaan we over tot het daadwerkelijk wijzigen van alle velden. Dit gebeurt, door voor alle velden nieuwe gegevens in te toetsen:

```
3040 PRINT CHR$(19) : GOSUB
520: REM PRINT HOME
3050 FOR II = 0 TO 5
3060 PRINT VN$(II);": ";
3070 INPUT HA$(II)
3080 NEXT II
3090 PRINT : PRINT "Korrekt in-
gevoerd (J/N)";
3100 GOSUB 850
3110 IF T$="n" OR T$ = "N" THEN
3000
```

Ook na de invoer weer een controle op de juistheid van de gegevens, waarna we overeenkomstig het antwoord ofwel opnieuw aan de invoer beginnen, ofwel het gewijzigde record in ons adressenbestand opnemen:

```
3120 AA=I : GOSUB 300 : REM
wegzetten in array
3130 PRINT:PRINT "Meer records
wijzigen (J/N)";
3140 GOSUB 850
3150 IF T$="n" OR T$ = "N" THEN
RETURN
3160 GOTO 3000
```

Al met al een niet erg moeilijk te begrijpen routine. Als je deze vergelijkt met de regels tussen 2000 en 300, waar een record wordt toegevoegd aan het bestand zul je weinig verschillen tegen komen. Het enige echte verschil zit in de indexvariabele van array A$(). Bij het toevoegen hebben we daar steeds de hoogste waarde, terwijl in dit geval de indexwaarde van de zoeklus (**I**) wordt gebruikt, om het juiste record te bepalen en bewerken. Daarom is met name regel 3120 sterk verschillend van regel 2120, hoewel dezelfde subroutine (300) wordt gebruikt. Bekijk deze module rustig en, wat meer is, probeer hem uit op je eigen computer.

Tot slot.

Hiermee hebben we weer twee leuke modules aan ons programma toegevoegd. We kunnen nu niet alleen adressen toevoegen aan het bestand, maar ook zoeken in alle mogelijke velden, terwijl we als laatste

ook wijzigingen in de bestaande adresrecords kunnen aanbrengen. Het geheel begint daardoor ook steeds meer op een volwassen programma te lijken. Voor de volledigheid zullen we het complete programma tot op dit moment in z'n geheel nog een keer afdrukken:

```
10REM Menu gestuurde DATABASE
20REM voor adressenbestand
30REM Commodore-Info 1987
40REM 870601/ v.1.04
100REM initialisatie
110REM
120TT=0 : REM totaalaantal records
130DIM A$(300,5) : REM aantal records max.
140KZ=0 : REM keuze
150I = 0 : REM temp.var
160DIM KZ$(10) : REM keuzestrings
170DIM ST((LOG(300)/LOG(2)+4) , 1) :REM TEMP-
ARRAYS
180DIM VN$(7) : REM veldnamen
200REM init
210GOSUB 900 : REM keuze lezen
299GOTO 1000 : REM begin programma
300REM var. wegzetten in array
310FOR II = 0 TO 5
320A$(AA,II)=HA$(II)
330NEXT
340RETURN
399REM data keuzemogelijkheden
400DATA "KEUZEMENU ADRESSENBESTAND"
410DATA "Records toevoegen"
411DATA "Records wijzigen"
412DATA "Records verwijderen"
413DATA "Records opzoeken"
414DATA "Bestand sorteren"
415DATA "Bestand printen"
416DATA "Bestand inlezen"
417DATA "Bestand wegschrijven"
418DATA "Stoppen"
419DATA "eind"
420DATA "Naam              "
421DATA "Voorletters       "
422DATA "Straat en nummer  "
423DATA "Postkode          "
424DATA "Woonplaats        "
425DATA "Telefoon          "
500REM printen kop
510PRINT CHR$(147) : REM CLS
520PRINT"**************************************"
530PRINT: PRINT TAB(20-LEN(KZ$
   (KZ))/2); KZ$(KZ):PRINT
540PRINT "Aantal records : ";TT
550PRINT"**************************************"
560PRINT
570RETURN
600REM menu printen en keuze
610GOSUB 500: REM printkop
620FOR I= 1 TO TK
630PRINT I;" - "; KZ$(I)
640NEXT I
650PRINT : PRINT "Maak een keuze ";
660GOSUB 800 : REM lees toets
670KZ= VAL(T$)
680IF KZ OR KZ  TK THEN 660
690RETURN : REM terug met kz
800REM inlezen van een toets
810T$ = INKEY$ : REM lees toets
820IF T$="" THEN 810
830RETURN : REM terug met t$
850REM JA/NEE input
852OK = 0 : REM vlag
855A$="NnJj": REM keuzemogelijkheden
860GOSUB 810 : REM input toets
865FOR II= 1 TO LEN(A$)
870IF T$ = MID$(A$,II,1)  THEN OK=1
880NEXT II
890IF OK THEN RETURN: REM met j/n
895GOTO 860 : REM opnieuw
898END
899REM inlezen keuzemogelijkheden
900I = 0 : TK=0
910READ KZ$(I)
920IF KZ$(I) = "eind" THEN TK = I - 1 : GOTO 950
930I = I + 1
940GOTO 910
950FOR I = 0 TO 5
960READ VN$(I)
970NEXT I
1000REM begin programma
1010SU=0:KZ=0:GOSUB 600 : REM menu
1020IF KZ = TK THEN 9500 : REM einde
1030ON KZ GOSUB
2000,3000,4000,5000,6000,7000,8000,9000
1040GOTO 1010 : REM opnieuw menu
2000REM toevoegen aan bestand
2010GOSUB 500 : REM printen kop
2020PRINT
2030FOR II = 0 TO 5
2040PRINT VN$(II);": ";
2050INPUT HA$(II)
2060NEXT II
2090PRINT : PRINT "Korrekt ingevoerd  (J/N)";
2100GOSUB 850
2110IF T$="n" OR T$ = "N" THEN 2000
2120TT=TT+1: AA=TT:GOSUB 300 :
    REM wegzetten in array
2130PRINT:PRINT "Meer records invoeren  (J/N)";
2140GOSUB 850
2150IF T$="n" OR T$ = "N" THEN RETURN
2160GOTO 2000
2999REM Muteren / Wijzigen
3000GOSUB 500: REM kopprint
3005SU=2: GOSUB 5010 :IF SU = 2 THEN RETURN
3010GOSUB 500: REM KOPPRINT
3011FOR II = 0 TO 5
3012PRINT VN$(II);": ";A$(I,II)
3014NEXT II
3020PRINT:PRINT "Wijzigen zeker  (J/N)"
3030GOSUB 850: IF T$="N" OR T$="n"
    THEN RETURN
3040PRINT CHR$(19) : GOSUB 520:
    REM PRINT HOME
3050FOR II = 0 TO 5
3060PRINT VN$(II);": ";
3070INPUT HA$(II)
3080NEXT II
3090PRINT : PRINT "Korrekt ingevoerd  (J/N)";
3100GOSUB 850
3110IF T$="n" OR T$ = "N" THEN 3000
3120AA=I : GOSUB 300 : REM wegzetten in array
3130PRINT:PRINT "Meer records wijzigen  (J/N)";
3140GOSUB 850
```

```
(vervolg)

3150IF T$="n" OR T$ = "N" THEN RETURN
3160GOTO 3000
3999REM verwijderen

6999REM ZOEKEN in bestand
7000GOSUB 500 : REM kopprint
7010PRINT:PRINT"Geef sleutelveld :"
7020FOR II = 0 TO 5
7030PRINT II+1;" - ";VN$(II)
7040NEXT II
7050PRINT : PRINT "Maak een keuze ";
7060GOSUB 800 : REM lees toets
7070KK= VAL(T$)
7080IF KK OR KK  7  THEN 7050
7090IF KK = 7 THEN RETURN:REM menu
7100VL=KK-1 : REM veldsoort
7110VLAG=0
7120GOSUB 500 : REM kopprint
7130PRINT:PRINT:PRINT
7140PRINT "zoeken op "; VN$(VL);" :";
7150INPUT Z$
7160FOR I = 1 TO TT : REM loop alle records
7170IF LEFT$(A$(I,VL),LEN(Z$)) = Z$
     THEN GOSUB 7210:IF SU=99 THEN I = TT
7180NEXT I
7190IF VLAG = 0 THEN 7380
7200RETURN
7210REM record gevonden - print
7220GOSUB 500
7230VLAG =1
7240PRINT "Record nummer : ";I
7250FOR II = 0 TO 5
7260PRINT VN$(II)" : "A$(I,II)
7270NEXT II : PRINT
7280ON SU+1 GOTO 7300, 7340, 7340 :
     REM SU=SOORT MODULE
7290REM SU=0 normaal zoeken
7300PRINT " Typ toets voor vervolg"
7310GOSUB 800
7320GOTO 7370
7330REM SU=1 Verwijderen SU=2 Muteren
7340PRINT "Dit record bewerken  (J/N)"
7350GOSUB 850
7360IF T$="J" OR T$="j" THEN SU=99
7370RETURN
7380REM record niet aanwezig
7390PRINT Z$ " niet gevonden"
7400PRINT "Typ toets voor vervolg"
7410GOSUB 800
7420RETURN
```

Tot zover het programma voor deze keer. We zijn al een heel eind op weg, om het een volwassen programma te laten worden. De volgende keer zullen we een leuke manier introduceren, om bepaalde adressen te kunnen verwijderen en het geheel te kunnen printen. Je kunt ook alvast zelf beginnen om het programma te completeren. Ik wens je sterkte en veel programmeerplezier.

Jan Bodzinga

De cursus machinetaal voorziet in een grote behoefte onder onze lezers. Niet alleen zijn we heel simpel begonnen, ook is het geheel voor de minder ervaren (machinetaal)-programmeur nog allemaal best te begrijpen. Tjipke van der Land doet z'n uiterste best om dat zo te houden, juist nu het allemaal wat ingewikkelder wordt.

Cursus Machinetaal

Deel 6: Instructieset

Na alle droge stof die we gehad hebben is het tijd om te beginnen met de instructie set voor de 6502 en aanverwante micro-processoren. De bedoeling is elke instructie afzonderlijk vast te pakken en vervolgens te ontleden met alle zaken die daar verder bij komen kijken. Er bestaan een hele serie instructies voor de Commodore processor.

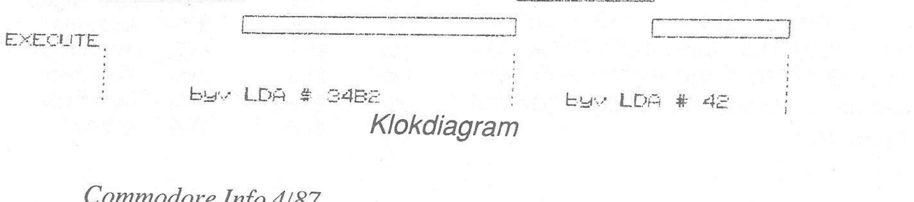
De eerste instructie die we gaan leren is de instructie **LDA** wat betekent LAAD ACCU. LDA is een **Mnemonic code**, en wat dat precies betekend zien we straks. Het laden van de accu, de accumulator of register A, kan op verschillende manieren worden gedaan. Je kunt bijvoorbeeld de accu direct laden met een getal, een **constante**, maar je kunt de accu ook laden met een getal wat hij nog ergens vanuit het computergeheugen moet ophalen. Op z'n *jan boerefluitjes* uit gelegd komt het ongeveer hier op neer: Op het moment dat je wat wilt weten uit een boek, kun je op verschillende manieren aan die informatie komen, je kunt dat boek je rechtstreeks laten aangeven , of je laten vertellen waar je het boek kunt vinden. Deze twee manieren worden in machinetaal **immediate** en **absolute** genoemd.
Immediate, engels voor onmiddellijk, is rechtstreeks een getal of waarde in de accu stoppen. Deze instructie neemt twee bytes in beslag, 1 byte voor de instructiecode (LDA) en 1 byte voor de operand (het getal). Als je hierover meer wilt weten, moet je de vorige les nog maar eens nalezen.
Op de **absolute** manier een getal binnen lezen betekent dat de accu verteld wordt waar hij z'n getal kan ophalen. Voor dit halen heeft de accu een (computer) geheugenadres nodig, en we weten inmiddels allemaal dat het adres 16 bits breed is, dus twee bytes.
Wat iedereen daarbij op z'n klompen kan aanvoelen is, dat deze instructie in totaal 3 bytes gebruikt. Pakken we de **instructieset** van de 65xx erbij, dan zien we achter de instructie LDA een hele rij van instructie methoden. Deze methoden zullen we niet allemaal in één keer behandelen, omdat we eerst wat meer eenvoudige instructies onder de knie moeten hebben om de andere adresseermethoden te kunnen begrijpen. Dat houdt echter niet in dat we het deze keer bij twee adresseermethoden laten.

Adresseren

We gaan in totaal drie adresseer-methodes behandelen, namelijk de hierboven genoemde twee methodes, met daar aan toegevoegd de **zero page** adresseer-methode. Om een machinetaal programma zo snel mogelijk te kunnen laten runnen, moet je het totaal programma uit zo weinig mogelijk instructies (en vooral weinig bytes) laten bestaan, omdat elke **instructie cyclus** een klokpuls in beslag neemt.
In het voorbeeld van het klokdiagram kun je zien dat voor elke byte die verplaatst wordt een klokpuls wordt gebruikt. Wat voor iedereen duidelijk zal zijn is, dat zodra er minder instructies zijn, ook minder tijd nodig is om het programma te doorlopen.
Als voorbeeld stel ik dat je aan het hardlopen bent op een route, met als bedoeling ook op z'n snelst, als eerste dus, over de finish te gaan. Op het moment dat je op de route een obstakel tegenkomt, probeer je normaal gesproken met zo weinig mogelijk tijdverlies dit obstakel te omzeilen. Deze mogelijkheden zijn af en toe ook toe te passen in machinetaal, alleen niet door over een steen te springen, maar door bijvoorbeeld 1 byte van een instructie weg te werken waardoor je bijvoorbeeld van 3 naar 2 bytes in een instructie gaat.
Wat je op zo'n moment gewonnen hebt is niet meer dan 1 microseconde, maar je hebt wel op dat moment 30% snelheidswinst.

Zero Page

Waar dit hele verhaal met hardlopers en obstakels naar toe leidt is uiteraard de **Zero page instructieme-**



Klokdiagram

thode. Deze instructie-methode is speciaal bedoeld voor de zogenaamde nul pagina. Een computergeheugen kunnen we bij de 65XX namelijk opdelen in **pagina's**. Wat ik me op dit moment kan voorstellen is dat je niet helemaal het verband ziet tussen een pagina en een geheugen!
Stel dat je nog een boek hebt waar in totaal 64000 bytes (letters) in zitten, die natuurlijk niet allemaal op 1 bladzijde kunnen staan. In de eerste plaats is dat niet uitvoerbaar, en in de tweede plaats wordt je erg moe van het lopen langs de regels op de bladzij om alle 64000 letters te kunnen lezen. De oplossing was om de hoeveelheid letters op te delen in meerdere bladzijden. In ons boek, de computer, stoppen we nu op elke bladzijde precies 256 bytes. Bij deze 256 wordt ook de 0 meegeteld. Dit wordt dus in principe van 0 tot 255.

Deze manier wordt in de computertechniek altijd toegepast, omdat het binaire stelsel ook rekent met 0 als een van z'n twee mogelijke waarden. (0 en 1). Deze consequentie voeren we dus maar even door in ons eigen machinetaalstelsel.
Na een kleine rekensom komen we snel tot de ontdekking dat ons boek uit precies 255 bladzijden moet bestaan.
Je zult inmiddels wel vermoeden dat dit voorbeeld een beetje overeenkomt met het **echte** computer geheugen, en dat is ook de bedoeling. Als we een beetje doorrekenen bijvoorbeeld door 255 decimaal om te rekenen naar hexadicimaal dan zien we dat **255 = $ FF**. Tot onze vreugdevolle ontdekking zien we dus dat dit precies een byte is, want $FF komt overeen met 8 bits vol enen (%1111 1111). Dit was juist de waarde die we wilden besparen op een instructie.

Wat we hieruit kunnen opmaken is dat je met 1 byte een hele pagina in het geheugen kunt adresseren. Als we dit wat duidelijker stellen, dan komt het er op neer, dat voor ieder adres (16 bits) twee bytes nodig zijn, waarvan het eerste byte (high byte) het paginanummer in het geheugen aangeeft, terwijl het tweede byte (low byte) de plaats van het adres binnen de pagina aan duidt. Daarop door bordurend, zien we, dat er in feite niets nodig is, om pagina NUL, de Zero Page van een adres te voorzien. We kunnen daar de high-byte gewoon laten vervallen, waardoor alle instructies, die naar de Zero Page wijzen een byte **kleiner** zijn, wat de nodige tijdwinst oplevert!

Instructieset 6500 serie

Machine Opcodes		Mnemonic Opcodes	Addressing Mode
Dec	Hex		
109	$6D	ADC	Absolute
125	$7D	ADC	Absolute,X
121	$79	ADC	Absolute,Y
105	$69	ADC	Immediate
097	$61	ADC	(Indirect,X)
113	$71	ADC	(Indirect),Y
101	$65	ADC	Zero Page
117	$75	ADC	Zero Page,X
045	$2D	AND	Absolute
061	$3D	AND	Absolute,X
057	$39	AND	Absolute,Y
041	$29	AND	Immediate
033	$21	AND	(Indirect,X)
049	$31	AND	(Indirect),Y
037	$25	AND	Zero Page
053	$35	AND	Zero Page,X
014	$0E	ASL	Absolute
030	$1E	ASL	Absolute,X
010	$0A	ASL	Accumulator
006	$06	ASL	Zero Page
022	$16	ASL	Zero Page,X
144	$90	BCC	Relative
176	$B0	BCS	Relative
240	$F0	BEQ	Relative
044	$2C	BIT	Absolute
036	$24	BIT	Zero Page
048	$30	BMI	Relative
208	$D0	BNE	Relative
016	$10	BPL	Relative
000	$00	BRK	Implied
080	$50	BVC	Relative
112	$70	BVS	Relative
024	$18	CLC	Implied
216	$D8	CLD	Implied
088	$58	CLI	Implied
184	$B8	CLV	Implied
205	$CD	CMP	Absolute
221	$DD	CMP	Absolute,X
217	$D9	CMP	Absolute,Y
201	$C9	CMP	Immediate
193	$C1	CMP	(Indirect,X)
209	$D1	CMP	(Indirect),Y
197	$C5	CMP	Zero Page
213	$D5	CMP	Zero Page,X
236	$EC	CPX	Absolute
224	$E0	CPX	Immediate
228	$E4	CPX	Zero Page
204	$CC	CPY	Absolute
192	$C0	CPY	Immediate
196	$C4	CPY	Zero Page
206	$CE	DEC	Absolute
222	$DE	DEC	Absolute,X
198	$C6	DEC	Zero Page
214	$D6	DEC	Zero Page,X
202	$CA	DEX	Implied
136	$88	DEY	Implied
077	$4D	EOR	Absolute
093	$5D	EOR	Absolute,X
089	$59	EOR	Absolute,Y
073	$49	EOR	Immediate
065	$41	EOR	(Indirect,X)
081	$51	EOR	(Indirect),Y
069	$45	EOR	Zero Page
085	$55	EOR	Zero Page,X
238	$EE	INC	Absolute
254	$FE	INC	Absolute,X
230	$E6	INC	Zero Page
246	$F6	INC	Zero Page
232	$E8	INX	Implied
200	$C8	INY	Implied
076	$4C	JMP	Absolute
108	$6C	JMP	(Indirect)
032	$20	JSR	Absolute
173	$AD	LDA	Absolute
189	$BD	LDA	Absolute,X
185	$B9	LDA	Absolute,Y
169	$A9	LDA	Immediate
161	$A1	LDA	(Indirect,X)
177	$B1	LDA	(Indirect),Y
165	$A5	LDA	Zero Page
181	$B5	LDA	Zero Page,X
174	$AE	LDX	Absolute
190	$BE	LDX	Absolute,Y
162	$A2	LDX	Immediate
166	$A6	LDX	Zero Page
182	$B6	LDX	Zero Page,Y
172	$AC	LDY	Absolute
188	$BC	LDY	Absolute,X
160	$A0	LDY	Immediate
164	$A4	LDY	Zero Page
180	$B4	LDY	Zero Page,X
078	$4E	LSR	Absolute
094	$5E	LSR	Absolute,X
074	$4A	LSR	Accumulator
070	$46	LSR	Zero Page
086	$56	LSR	Zero Page,X
234	$EA	NOP	Implied
013	$0D	ORA	Absolute
029	$1D	ORA	Absolute,X
025	$19	ORA	Absolute,Y
009	$09	ORA	Immediate
001	$01	ORA	(Indirect,X)
017	$11	ORA	(Indirect),Y
005	$05	ORA	Zero Page
021	$15	ORA	Zero Page,X
072	$48	PHA	Implied
008	$08	PHP	Implied
104	$68	PLA	Implied
040	$28	PLP	Implied
046	$2E	ROL	Absolute
062	$3E	ROL	Absolute,X
042	$2A	ROL	Accumulator
038	$26	ROL	Zero Page
054	$36	ROL	Zero Page,X
110	$6E	ROR	Absolute
126	$7E	ROR	Absolute,X
106	$6A	ROR	Accumulator
102	$66	ROR	Zero Page
118	$76	ROR	Zero Page,X
064	$40	RTI	Implied
096	$60	RTS	Implied
237	$ED	SBC	Absolute
253	$FD	SBC	Absolute,X
249	$F9	SBC	Absolute,Y
233	$E9	SBC	Immediate
225	$E1	SBC	(Indirect,X)
241	$F1	SBC	(Indirect),Y
229	$E5	SBC	Zero Page
245	$F5	SBC	Zero Page,X
056	$38	SEC	Implied
248	$F8	SED	Implied
120	$78	SEI	Implied
141	$8D	STA	Absolute
157	$9D	STA	Absolute,X
153	$99	STA	Absolute,Y
129	$81	STA	(Indirect,X)
145	$91	STA	(Indirect),Y
133	$85	STA	Zero Page
149	$95	STA	Zero Page,X
142	$8E	STX	Absolute
134	$86	STX	Zero Page
150	$96	STX	Zero Page,Y
140	$8C	STY	Absolute

Machine Opcodes		Mnemonic Opcodes	Addressing Mode
Dec	Hex		
132	$84	STY	Zero Page
148	$94	STY	Zero Page,X
170	$AA	TAX	Implied
168	$A8	TAY	Implied
186	$BA	TSX	Implied
138	$8A	TXA	Implied
154	$9A	TXS	Implied
152	$98	TYA	Implied

Zoals eerder verteld staat bij nagenoeg iedere instructie een aantal **instructiemethoden**. Voor elke instructiemethode is een andere opcode. Een **opcode** is het hexadecimale getal waar de computer aan kan zien met wat voor type instructie hij te maken heeft. Zo'n opcode is altijd één byte lang, en elke andere instructie met de daaraan verbonden instructie-methode heeft een andere code. Laten we nog maar eens even de instructie LDA pakken. Als we volgens de immediate methode wat willen uitvoeren dan is de opcode **$ A9**. Willen met deze zelfde instructie op de absolute methode iets uitvoeren dan is de opcode **$ AD**, in Zero page Methode is dit **$ A5**.

Opcodes

Waar het nu om gaat is dat de processor aan de opcode precies kan zien wat hij moet gaan doen.
Als voorbeeld zullen we een getal $ 23 nemen wat op geheugenplaats $ 345C staat. Als eerste gaan we dit getal rechtstreeks in de accu, register A, laden, dus een immediate instructie. Deze ziet er dan als volgt uit:

$ A9 23LDA $23

Nu gaan we beredeneren hoe de microprocessor deze instructie verwerkt. Bij het ophalen van de eerste waarde , wat dus A9 is, gaat de microprocessor decoderen (ontleden) welke instructie methode dit is. Door z'n eigen aanwezige microprogramma (zie deel 3 van deze cursus) weet hij zelf dat hij de accu moet laden volgens de immediate methode, dus weet hij dat de waarde die hij moet laden gelijk achter de instructie staat, wat in dit geval dus $ 23 is.

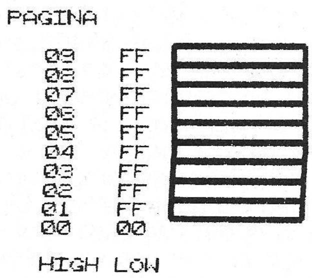
De volgende stap is de microprocessor te volgen bij de verwerking volgens de absolute methode. Hier dus weer hetzelfde verhaal wat betreft het ophalen van het eerste getal, waar hij dus gaat kijken welke soort instructie het is. Hier weet de microprocessor dat het getal wat achter de instructie staat een **adres** moet zijn waarvan hij ook weet dat dit totaal 2 bytes moeten zijn. (16 bits dus 2 bytes). Het enige wat enigszins van onze logische gedachtengang afwijkt is dat de processor z'n adres **achterste voren** ophaalt. Waarom hij dat doet zal straks wel duidelijk worden.
Dus de processor haalt eerst zijn zogenaamde lowbyte op en daarna het highbyte. De totaal instructie ziet er dan als volgt uit:

$ AD 5C 34LDA $345C

Dus aan de waarde AD weet de processor dat hij een adres moet halen. Zodra hij het adres binnen heeft adresseert hij met deze waarde het geheugen en haalt het getal op wat op het aangegeven adres staat.
Om nu er nog even op terug te komen waarom de processor zijn adres omgekeerd binnenhaalt, is goed uit te leggen met de zero page instructie methode. Bij deze methode is namelijk van hetzelfde sprake als bij de absolute methode alleen hier hoeft de processor niet het high byte op te halen, want hij weet dat hij op de nul pagina zijn waarde moet gaan halen. Dus de enige waarde waar hij mee moet rekenen is het **low byte**.
Stel dat de waarde 23 op geheugen plaats $ 00 86 staat, dan ziet de instructie er als volgt uit:

$ A5 86LDA $86

Je kunt dan ook ogenblikkelijk zien waarom het zinnig is om eerst het low byte te halen en daarna het high byte, want als we nu op onze logische manier het adres gingen ophalen, dan zouden we in de moeilijkheden raken, want dan zag de totaal instructie er zo uit : **A5 00 86**, waardoor het nut van de zero page adressing te niet wordt gedaan. Dus met de totaal instructie **A5 86** weet de microprocessor dat hij een getal moet halen van adres **$ 0086**, want door de opcode A5 weet de processor dat het de nul pagina is, dus het high byte is dan $ 00 en hoeft daarom niet te worden vermeld.



voorbeeld van pagina's in geheugen

Mnemonics

Het zal nu duidelijk zijn wat het verschil is tussen de verschillende **adresseer methoden**. Een aantal lessen terug hebben we verteld dat we het hexadecimale stelsel gingen toepassen omdat dit leesbaarder zou zijn dan binair. Dit kunnen we ook nog een keer doorvoeren op hexadecimaal niveau als we over instructies spreken, want de de opcode's A9, AD en A5 hebben alle drie betrekking op het laden van de accu. Om dit beter leesbaar te maken is de benaming **mnemonic** in het leven geroepen, zoals eerder genoemd,. Met dit mnemonic krijgen we in verloop van de cursus nog veel te maken, dus zullen we dit even grondig doornemen. Mnemonics komen we veel tegen als we gaan werken met de zogenaamde Assembler programma's.

Assembler

Een Assembler is een stukje hulpgereedschap, meestal software, wat we gebruiken om sneller in machinetaal te kunnen schrijven, want als je een stukje machinetaal moet schrijven door elke keer een reeks hexadecimale getallen achter elkaar te zetten, dan zit de kans er in dat je veel fouten maakt en bovendien gaat het heel erg langzaam (het programmeren wel te verstaan).

De mnemonic-code voor het laden van de accu is, zoals al eerder genoemd, **LDA**. Het LDA is dan uiteraard onder te verdelen in de Immediate en de Absolute methode's. We zullen zo dadelijk een overzichtelijk tabelletje maken, zodat je het zoëven geleerde gedeelte kunt overzien. Hoe het programmeren in assembler gaat, behandelen we nu nog niet want het is veel zinniger eerst te begrijpen hoe het programmeren op hexadecimaal niveau gaat. Waar we gelijk wel mee te maken krijgen is uiteraard de mnemonic code, omdat deze in de instructie boekjes worden gebruikt. Als voorbeeld zullen we een getal $ 23 nemen dat staat op adres $ 005C, die we volgens de drie behandelde adresseer manieren zullen binnenhalen:

MNEMONIC, IMMEDIATE-waarde
<u>ABSOLUTE-adres, ZERO PAGE-adres</u>

LDA A9 23 AD 5C 00 A5 5C

Bij *immediate* zien we dat het juiste getal gelijk wordt binnen gelezen in de accu, en bij *absolute* zien we dat

het gehele adres wordt binnen gehaald om de lokatie aan te wijzen waar het getal te lezen valt. Bij de *Zero page* methode zien we precies hetzelfde als bij absolute, alleen wordt hier in dit geval een byte aan tijd en hoeveelheid bespaard. In dit geval kan je wel zero-page adressing toepassen in de plaats van absolute, maar zodra er sprake is van adressen hoger dan de **nul pagina** dan kan dit niet meer.

Poke & peek

Nu je dit allemaal een beetje duidelijk is zal je zeggen OK dit is allemaal prachtig, maar hoe kan ik dat in de Computer stoppen. Aangezien dit wel het belangrijkste is, zullen we daar in dit deel ook nog wat aan doen, wel te verstaan dat je pas echt kan zien wat je doet in de volgende les. Even geduld hebben dus nog. Wat we nu wel gaan behandelen is hoe je dit echt laat werken in de computer. Voordat we dit gaan aansnijden gaan we eerst praten over **poken** en **peeken**, omdat dat op dit moment nog de enige manier is om op machinetaal niveau met de computer de communiceren. Straks met de machinetaalmonitor en de assembler kunnen we rechtstreeks zien wat we doen, maar nu nog niet. Met **poken** kunnen we rechtstreeks vanuit Basic waardes in het geheugen plaatsen. Deze waardes kunnen dus bijvoorbeeld opcode's en operands zijn, maar ook data ten behoeve van het programma. Dit vereist misschien enige uitleg, omdat we er nog niet over hebben gesproken. We hebben het inmiddels wel gehad over het ophalen van getallen die geladen moesten worden in de accu, maar nog niet over het programma zelf, dat uiteraard ook ergens in het geheugen draait. We onderscheiden in het **geheugen** dus een plek waar het programma loopt en een plek waar we **data** heen schrijven en weer ophalen. Deze plekken kunnen best een eind uit elkaar liggen.
Wat we straks gaan doen is het programma **poken** op een plek in het geheugen, en ergens anders in het geheugen poken we data. Het is op dit moment jammer, dat we straks wel iets kunnen maken wat loopt, maar niet echt kunnen zien dat het werkt. Maar ook dat komt de volgende keer.

Poken wil zeggen: langs van te voren afgesproken regels in decimale vorm een adres aanwijzen waar een getal of waarde moet worden geplaatst. Stel dat we het getal $ 23 op geheugenplaats $ 0600 willen schrijven, dan doen we dit door eerst deze getallen om te zetten in decimale getallen, dus $ 23 word 35 en $ 0600 wordt 1536. De instructie voor Basic wordt 'poke adres,waarde' dus **poke 1536,35** . Op deze manier kun je dus een heel programma in het geheugen poken. Je ziet dat nog al veel in de listingrubriek, waar hele stukken machinetaal als DATA in de Basicprogramma's voorkomen. Door gebruik te maken van een loop en een read-opdracht worden al die data-getallen in het RAM-geheugen weggezet. Dit doen wij echter alleen in deze en de volgende uitgave, omdat anders heel snel de lol eraf is om in machinetaal te programmeren. Maar voor een goed begrip moet je het toch even gedaan hebben op deze manier.

Break

Voor we ons programma gaan poken is er eerst nog een instructie die we moeten weten, namelijk de **BREAK** instructie die we aan het eind van een programma moeten plaatsen, omdat de processor anders stomweg de instructies gaat uitvoeren die na het programma staan, wat dus meestal willekeurige waarden kunnen zijn. Het gevolg is, als we een programma laten lopen tot aan het eind, dat hij vast loopt en niet terug in Basic komt. Dit vastlopen kan komen omdat hij misschien opdrachten moet uitvoeren die niet erg logisch zijn, of hij blijft eeuwig in een loop staan draaien. De enige remedie is dan de computer uit en aan zetten met als gevolg dat het moeizaam ingevoerde programma ook helemaal opgelost is. Om dit te voorkomen zetten we achter elk programma de zogenaamde BREAK-instructie wat in hexadecimaal **$ 00** is. Deze BREAK instructie noemen ze de **IMPLIED** instructie methode, waar we later op terug komen.
Met deze reddende instructie komt de computer weer terug in Basic, en kun je weer gewoon doorwerken. Met deze kennis in ons achterhoofd gaan we even een klein programmaatje maken en in het geheugen poken, puur om toch onze reeds verworven kennis eens in de computer te stoppen. Stel dat we een getal $ 23 hebben op geheugenplaats $3400, en we willen dit getal laden in de accu. We laten het programma starten op $ 0600. In principe begin je altijd door een *flowchart* te maken. Dit is een globale schets hoe het programma in grote lijnen zal gaan werken. Bij het voorbeeld is dat uiteraard heel simpel, omdat we hier nog maar één instructie toepassen. We werken als volgt:

Gegevens: Een getal $ 23 staat op geheugenplek $ 3400;
Het getal moet binnen gehaald worden in de accu;

In dit geval moeten we eerst zelf het getal $ 23 op de juiste plaats zetten, omdat we geen programma hebben lopen wat dat er voor ons neerzet. Bijvoorbeeld door het uit voeren van berekeningen.
Dus type in : **poke 13312,35**. Als je dit gedaan hebt dan staat er op geheugen-lokatie $ 3400 een waarde van $ 23. Nu moeten we dus nog het hele programma poken, wat het getal in de accu laadt. In mnemonic ziet dit er zo uit.

Mnemonic Hexadecimaal

LDA $ 3400 (absolute) AD 00 34
BRK 00

Je ziet dat ons programma nu uit totaal 4 bytes bestaat, wat nog te poken is, zonder last te krijgen van de wel bekende zweetdruppels. We hadden bepaald dat het programma zou starten op adres $ 0600, en moeten we het volgende weer intypen.

Opcode/operand **POKE ADRES, OPCODE/OPERAND**

AD OP $ 0600 POKE 1536,173
00 OP $ 0601 POKE 1537,0
34 OP $ 0602 POKE 1538,52
00 OP $ 0603 POKE 1539,0

Voordat je dit overneemt is het een goede oefening om het eerst zelf proberen te berekenen, wat je inmiddels wel moet kunnen. Nadat dit met succes is gedaan moeten we het programma alleen nog even laten runnen, zoals dat heet. Het runnen van een machinetaal programma in Basic doen we door middel van het **SYS commando**. Dit SYS commando verwacht evenwel ook een decimaal nummer, wat inmiddels wel bij ons bekend is namelijk 1536, het startadres van ons programma. Dus je kunt het programma starten door in te tikken **SYS (1536)**. Als je dat hebt ingetypt gaat hij de instructies vanaf $ 0600 uitvoeren.

Waarschijnlijk denk je "hij doet niks", maar dat komt omdat het zo snel gegaan is. Komt hij terug in Basic, dan heeft hij het toch goed gedaan, je kunt alleen niet zien hoe. Maar dat komt allemaal de volgende keer!
We gaan dan door met de volgende instructie, waarin o.a. staat hoe je door middel van een peek-instructie wél kunt zien wat je gedaan hebt. Alvast sterkte en tot de volgende keer.

Tjipke van de Land ●

Print-out

Met o.a. Blokmonitor – Mastermind – snake – Adresbak 128 – lijn 128 – lichtkrant C-16

Listingtelefoon

Voor vragen over de in Commodore Info verschenen listings blijft de listingtelefoon beschikbaar. Elke maandag zitten we van 17.00 tot 21.00 uur paraat voor vragen en opmerkingen op nummer 02155-25162. Denk er wel aan dat het niet mogelijk is om op andere tijden informatie over listings te krijgen, niet op bovenstaand nummer en ook niet bij de redactie.

Rob Goudriaan.

Inhoud van dit listingdeel

Syntax Checksum

Het overtikken van een listing kan een heel karwei zijn en als u een beetje normaal mens bent dan maakt u daarin beslist een aantal fouten. Nu is niets moeilijker om de fouten uit je eigen werk te halen. Al geruime tijd geleden heeft Jan Bodzinga hiervoor een zgn. **Checksum-programma** geschreven. Om de vele nieuwe lezers van Commodore-Info te helpen volgt hieronder nog een keer een volledige uitlegh over de werking van dit programma, waarmee het, hoe vreemd dat misschien ook lijkt, echt mogelijk is om met behulp van dit programma de fouten in elke door ons geplaatste listing op te sporen.
Hiervoor gaat u als volgt te werk:

❶ U tikt de listing heel zorgvuldig over en SAVEt hem voordat u het programma RUNt op een diskette of een cassette.

❷ U tikt het RUN commando in. Mocht het programma de boodschap **'Fout in dataregels!'** geven dan heeft u een fout bij het overtikken gemaakt. Herstel de fout en SAVE de verbeterde versie. Mocht het programma met de boodschap **'data is weggezet checksum testten met sys ....'** komen dan is tot dusver alles goed. Het programma is nu in een stukje machinetaalgeheugen gezet. Als u het NEW commando geeft blijft het toch in de computer staan.

Alle door ons geplaatste programma's zijn in Basic geschreven. Als u een programma heeft overgetikt SAVE het eerst, mocht er iets mis gaan dan hoeft niet de gehele listing opnieuw te gaan intikken. Als u nu een programma op fouten wilt gaan controleren dan kunt u dat in het geheugen laden (wel eerst het checksumprogramma hebben gerund). Vervolgens typt u zonder het programma te runnen de opdracht **sys 49152** (C-64) of **sys 1536** (C-16 en Plus/4) in.
Als alles goed is gegaan loopt er nu een rij regelnummers over het scherm met getallen erachter. Dezelfde lijst staat ook achter elk door ons geplaatste programma. Wijkt nu een nummer achter een regelnummer af van het nummer dat in het blad staat dan heeft u in die regel iets anders ingetikt dan er in het blad stond. U kunt de stroom getallen d.m.v. de RUN/STOP toets pauzeren en weer vervolgen met de F1 of F7 toets. Het is uitermate belangrijk dat u goed met dit programma overweg kunt en mocht U het niet goed werkend krijgen bel dan gerust even met onze listingservice telefoonlijn.

```
  1 rem ********************************
    **
  2 rem basic loader "SYNTAX.CHECKSUM"
  3 rem na de commando's 'run' en 'new'
  4 rem blijft dit programma in het ge-
  5 rem heugen. laad het te testen pro-
  6 rem gramma en tik daarna sys 49152.
  7 rem ********************************
    **
 10 i=49152  :rem beginadres
 20 reada:ifa<0then40:rem data ingeleze
    n
 30 pokei,a:i=i+1:b=b+a:goto20
 40 if b<>16844thenprint"[CLR-HOME]fout
     in dataregels!":b=0:end
 50 poke49184,148:poke49185,192
 55 i=49300
 60 read a: ifa<0then80
 70 pokei,a:b=a+b:i=i+1:goto60
 80 if b<>20068thenprint"[CLR-HOME]fout
     in dataregels! (vanaf regel 240)":
    b=0:end
 90 print"data is weggezet"
 95 print"checksum testen met sys49152"
100 data 165,43,166,44,133,163,134,164,
     169, 147
110 data 32,210,255,160,0,240,3,32,73,
     192
120 data 32,73,192,208,1,96,32,225,255,
      208
130 data 3,76,116,164,32,81,192,32,73,1
    92
140 data 240,12,201,32,240,247,24,101,1
    67,133
150 data 167,76,37,192,166,167,169,0,13
    2,168
160 data 32,205,189,169,13,32,210,255,1
    64, 168
170 data 76,17,192,200,208,2,230,164,17
    7,163
180 data 96,162,0,189,123,192,240,6,32,
    210
190 data 255,232,208,245,32,73,192,170,
    32,73
200 data 192,132,168,32,205,189,162,3,1
    69,32
210 data 32,210,255,202,208,250,169,0,1
    33,167
220 data 164,168,96,82,69,71,69,76,32,0
230 data -1
240 data 165,197,201,3,240,7,201,4,240
250 data 6,76,148,192,76,34,192,169
260 data 147,32,210,255,76,161,192
270 data -1

*** EINDE LISTING ***

   syntaxchecksum listtestprogramma

   regel 1    249
   regel 2    84
   regel 3    125
   regel 4    2
   regel 5    246
   regel 6    152
   regel 7    249
   regel 10    157
   regel 20    64
   regel 30    38
   regel 40    57
   regel 50    14
   regel 55    251
   regel 60    192
   regel 70    42
   regel 80    244
   regel 90    245
   regel 95    237
   regel 100    183
   regel 110    158
   regel 120    232
   regel 130    183
   regel 140    96
   regel 150    96
   regel 160    127
   regel 170    71
   regel 180    223
   regel 190    73
   regel 200    79
   regel 210    109
   regel 220    106
   regel 230    225
   regel 240    16
   regel 250    163
   regel 260    92
   regel 270    225

ready.
```

Blokmonitor

Voor de wat meer gevorderde programeurs schreef Marcel de Groot een programma dat je in saat stelt om elke willekeurige track en sector op de schijf te lezen, ja zelfs direkt hierop te wijzigen. Met de funktie toetsen kan men kiezen of men wil bladeren, een ander blok wil inlezen, gaan editten of wegschrijven.

```
  1 rem blok-monitor / cbm-64 & 1541
  2 rem door marcel de groot
  3 rem uit hillegom / 02520-16616
  4 rem
 10 hx$="0123456789abcdef":dimhx$(255)
    ,d$(255):z$="[HOME][16xCRSR-DOWN]"
    :poke649,1
 20 sp$="[26xSPACE]":v$=">druk[SPACE]o
    p[SPACE]een[SPACE]toets<":r$=chr$(
    13)
 30 w$="[SHIFT CLR]"+sp$+"[2xSPACE][CT
    RL 2]B[CRSR-DOWN][CRSR-LEFT]B[CRSR
    -DOWN][CRSR-LEFT]B[CRSR-DOWN][CRSR
    -LEFT]B[CRSR-DOWN][CRSR-LEFT]B[CRS
    R-DOWN][CRSR-LEFT]B[CRSR-DOWN][CRS
    R-LEFT]B[CRSR-DOWN][CRSR-LEFT]B[CR
    SR-DOWN][CRSR-LEFT]B[CRSR-DOWN][CR
    SR-LEFT]B[CRSR-DOWN][CRSR-LEFT]B[C
    RSR-DOWN][CRSR-LEFT]B[CRSR-DOWN][C
    RSR-LEFT]B":n$=chr$(0)
 40 w$=w$+"[CRSR-DOWN][CRSR-LEFT]B[CRS
    R-DOWN][CRSR-LEFT]B[CRSR-DOWN][CRS
    R-LEFT]B"+r$+"CCCCCCCCCCCCCCCCCCCC
    CCCCCCCC[COM E]CCCCCCCCCCC":f$=sp$
    +r$
 50 fori=1to40:t$=t$+"C":next:w$=w$+"[
    2xCRSR-DOWN]"+t$+chr$(8)+chr$(142)
    :poke53280,6
 60 fori=1to16:forj=1to16:hx$(a)=mid$(
    hx$,i,1)+mid$(hx$,j,1):a=a+1:next:
    next
 70 poke53281,6:q$=z$+"[2xCRSR-DOWN][1
    3xCRSR-RIGHT]-"+sp$+"[CRSR-DOWN]"+
    f$+f$+f$+sp$
 80 printw$z$"[CRSR-DOWN]track?[SPACE]
    ";:poke198,0:g$="":gosub470:t=val(
    g$):t$=g$:v=0
 90 print"[4xSPACE]sector?[SPACE]";:g$
    ="":gosub470:s=val(g$):printr$"[CR
    SR-UP]"sp$:ift>35then80
100 ift>30ands>16ort>24ands>17ort>17an
    ds>18ors>20ort=0then80
110 printz$"[CRSR-DOWN]track:[SPACE][C
    TRL 9]"t$"[CTRL 0][4xSPACE]sector:
    [SPACE][CTRL 9]"g$
120 gosub410:ifa$<>"00"thenprintz$"[4x
    CRSR-DOWN]"v$:poke198,0:wait198,1:
    goto80
130 printq$z$"[4xCRSR-DOWN][CTRL 9]f1[
    CTRL 0][SPACE]=[SPACE]andere[SPACE
    ]pagina"r$"[CTRL 9]f3[CTRL 0][SPAC
    E]=[SPACE]ander[SPACE]blok[SPACE]i
    nlezen"
140 print"[CTRL 9]f5[CTRL 0][SPACE]=[S
    PACE]edit[SPACE]blok"r$"[CTRL 9]f7
    [CTRL 0][SPACE]=[SPACE]blok[SPACE]
    terugschrijven"
150 poke198,0:wait198,1:geta$:ifa$="[F
    1]"thenv=16-v:printq$"[HOME]";:gos
    ub430:goto130
160 ifa$="[F3]"then80
170 ifa$="[F5]"then260
180 ifa$<>"[F7]"then130
190 printq$z$"[4xCRSR-DOWN]blok[SPACE]
    terugschrijven[SPACE](j/n)?"
200 geta$:ifa$="n"then130
210 ifa$<>"j"then200
220 printq$:open1,8,15,"i":gosub460:if
    a$<>"00"thenclose1:goto250
230 open2,8,2,"#":print#1,"b-p[SPACE]2
    [SPACE]0":fori=0to255:print#2,left
    $(d$(i)+n$,1);:next
240 print#1,"u2[SPACE]2[SPACE]0";t;s:c
    lose2:gosub460:print#1,"i":close1:
    ifa$="00"then130
250 printz$"[4xCRSR-DOWN]opnieuw[SPACE
    ]proberen[SPACE](j/n)?":goto200
260 r=0:h=0:printq$z$"[4xCRSR-DOWN]bew
    eeg[SPACE]met[SPACE]cursor-toetsen
    "
270 print"[CRSR-DOWN][CTRL 9]f1[CTRL 0
    ][SPACE]=[SPACE]andere[SPACE]pagin
    a"r$"[CTRL 9]f7[CTRL 0][SPACE]=[SP
    ACE]stoppen":poke198,0
280 l=1028+int(r/8)*40+(rand7)*3+h:pok
    el,peek(l)or128:wait198,1
290 geta$:pokel,peek(l)and127:ifa$="[C
    RSR-DOWN]"andr<120thenr=r+8:goto28
    0
300 ifa$="[CRSR-UP]"andr>7thenr=r-8:go
    to280
310 ifa$="[CRSR-RIGHT]"and((rand7)<7or
    h=0)thenh=1-h:r=r+1-h:goto280
320 ifa$="[CRSR-LEFT]"and((rand7)>0orh
    =1)thenr=r-1+h:h=1-h:goto280
330 ifa$="[HOME]"thenr=0:h=0:goto280
340 ifa$="[F1]"thenv=16-v:printq$"[HOM
    E]";:gosub430:goto260
350 ifa$="[F7]"thenprintq$:goto130
360 a=asc(a$)-48:ifa<0ora>22or(a>9anda
    <17)then280
370 ifa>9thena=a-7
380 j=r+8*v:d$(j)=chr$((asc(d$(j)+n$)a
    nd15*16↑h)+a*16↑(1-h)):pokel,asc(a
    $)and63
390 h=1-h:b$=d$(j):if(asc(b$+n$)and127
    )<32thenb$="."
400 printleft$(z$,r/8+1)spc(31+(rand7)
    )b$:r=(r+1-h)and127:goto280
410 open1,8,15,"i":open2,8,2,"#":gosub
    460::ifa$<>"00"thenclose1:close2:r
    eturn
420 print#1,"u1[SPACE]2[SPACE]0";t;s:f
    ori=0to255:get#2,d$(i):next:gosub4
    60:close1:close2
430 fori=vtov+15:printhx$(i*8)":[SPACE
    ]";:forj=0to7:printhx$(asc(d$(i*8+
    j)+n$))"[SPACE]";
440 b$=d$(i*8+j):if(asc(b$+n$)and127)<
    32thenb$="."
450 l$=l$+b$:next:print"B[SPACE]'"l$"'
    ";:l$="":next:print:return
460 input#1,a$,b$,c$,d$:printz$"[2xCRS
```

```
    R-DOWN]disk-status:[SPACE][CTRL 9]
    "a$","b$","c$","d$r$"[HOME]";:retu
    rn
470 print"[CTRL 9]*[CTRL O][CRSR-LEFT]
    ";:wait198,1:geta$:a=asc(a$):ifa=1
    3andlen(g$)=2thenreturn
480 ifa=20andlen(g$)>0thenprint"[SPACE
    ][2xCRSR-LEFT]";:g$=left$(g$,len(g
    $)-1):goto470
490 ifa>47anda<58andlen(g$)<2theng$=g$
    +a$:printa$;
500 goto470
```

REGEL 1	149	REGEL 240	46
REGEL 2	139	REGEL 250	207
REGEL 3	43	REGEL 260	168
REGEL 4	143	REGEL 270	181
REGEL 10	240	REGEL 280	130
REGEL 20	251	REGEL 290	171
REGEL 30	16	REGEL 300	157
REGEL 40	151	REGEL 310	208
REGEL 50	154	REGEL 320	72
REGEL 60	187	REGEL 330	149
REGEL 70	174	REGEL 340	21
REGEL 80	164	REGEL 350	122
REGEL 90	10	REGEL 360	237
REGEL 100	135	REGEL 370	115
REGEL 110	2	REGEL 380	220
REGEL 120	217	REGEL 390	180
REGEL 130	134	REGEL 400	143
REGEL 140	250	REGEL 410	248
REGEL 150	235	REGEL 420	246
REGEL 160	123	REGEL 430	224
REGEL 170	172	REGEL 440	51
REGEL 180	91	REGEL 450	107
REGEL 190	198	REGEL 460	248
REGEL 200	175	REGEL 470	154
REGEL 210	27	REGEL 480	168
REGEL 220	44	REGEL 490	188
REGEL 230	155	REGEL 500	36

Mastermind

De vorige maand heeft er een listing van mastermind voor de c 16 in ons blad gestaan. Op veel verzoek nu een zelfde programma voor de c 64. Het programma heeft geen uitleg meer nodig, probeer de kleurcombinaties in zo min mogelijk beurten te raden. De moeilijkheidsgraad kan ingesteld worden. Maak het uzelf zeker in het begin niet te moeilijk.

```
 10 rem mastermind v1.17
 20 rem door jan pijnacker voor c'info
 30 rem zondag 29 juni 1986
 40 rem --- begin programma ---
 50 poke53280,0:poke53281,0:printchr$(
    142)chr$(8):s=54272:pokes+24,15
 60 fort=0to23:pokes+t,0:next
 70 cc$="[SHIFT CLR][5xSPACE][CTRL 9][
    CTRL 5][31xSPACE][CTRL O][5xSPACE]
    "
 80 sc$=cc$+"[4xSPACE][CTRL 9][CTRL 5]
    [SPACE][COM 3]mastermind[SPACE]doo
    r[SPACE]jan[SPACE]pijnacker[CTRL 5
    ][SPACE][CTRL O][4xSPACE]"
 90 sc$=sc$+mid$(cc$,2,38)
100 kl$="[CTRL 3][CTRL 4][CTRL 5][CTRL
     6][CTRL 7][CTRL 8][COM 3][COM 6]"
110 gosub800:a$="12345678"
120 be=1:gosub930:gosub440
129 rem bepaal kleuren
130 fora=0to3:b$(a)=mid$(a$,rnd(1)*8+1
    ,1):ifll=0anda<>0then890
140 tt$=tt$+mid$(kl$,val(b$(a)),1)+b$(
    a)
150 nexta
160 print"[HOME][6xCRSR-DOWN][4xCRSR-R
    IGHT]";:ti$="000000"
169 rem --- hoofd programma ---
170 gosub740:forb=0to3:poke198,0
180 getc$(b):ifc$(b)=""then180
190 ifc$(b)="←"thenprint:print"[CRSR-U
    P][4xCRSR-RIGHT][4xSPACE][4xCRSR-L
    EFT]";:fori=0to3:c$(i)="":nexti:go
    to170
200 ifc$(b)<"1"orc$(b)>"8"then180
210 print"[CTRL 9]"mid$(kl$,val(c$(b))
    ,1)c$(b);:nextb:print"[2xCRSR-RIGH
    T][CTRL O]";
220 forf=0to3
230 ifc$(f)=b$(f)thenprint"[COM 2]z";:
    a(f)=1:b(f)=1:aa=aa+1
240 nextf
250 ifaa=4then350
260 forc=0to3:ifa(c)=1thennextc:goto31
    0
270 ford=0to3
280 ifc$(c)=b$(d)andc<>dandb(d)=0thenp
    rint"[CTRL 2]w";:b(d)=1:goto300
290 nextd
300 nextc
310 fore=0to3:a(e)=0:b(e)=0:nexte:aa=0
320 print:print"[4xCRSR-RIGHT]";:be=be
    +1
330 ifbe=14thengosub630
340 goto170
349 rem alles goed
350 gosub770:gosub590:pokes+4,128:prin
    tsc$:print"[2xCRSR-DOWN]";:gosub69
    0
360 print"[HOME][6xCRSR-DOWN][2xCRSR-R
    IGHT][COM 8]alles[SPACE]goed[SPACE
    ]!!"
370 print"[CRSR-DOWN][2xCRSR-RIGHT]in"
    be"beurten
380 print"[CRSR-DOWN][2xCRSR-RIGHT]je[
    SPACE]deed[SPACE]er"int(ti/60)-2"s
    econden[SPACE]over"
390 print"[CRSR-DOWN][5xCRSR-RIGHT][CT
    RL 6][COM D]";:gosub910:print"[COM
     F][CRSR-DOWN][CRSR-LEFT]""[COM K]
    [CRSR-UP]"
400 print"[5xCRSR-RIGHT][CTRL 9][COM K
    ][CRSR-DOWN][CRSR-LEFT][CTRL O][CO
    M C][CTRL 9]";:gosub910:print"[CTR
    L O][COM V]"
410 print"[2xCRSR-UP][6xCRSR-RIGHT][CT
    RL 9][COM 6]druk[SPACE]'f1'[SPACE]
    voor[SPACE]nog[SPACE]een[SPACE]spe
    l":poke198,0
```

```
420 geta$:ifa$<>"[F1]"then420
430 run
439 rem scherm kleuren
440 print"[HOME][4xCRSR-DOWN][5xCRSR-R
    IGHT]"le$(-ll):t=20
450 printtab(t)"[CTRL 3][CTRL 9][SPACE
    ]1[SPACE]=>[SPACE]rood[9xSPACE]"
460 printtab(t)"[CTRL 4][CTRL 9][SPACE
    ]2[SPACE]=>[SPACE]cyaan[8xSPACE]"
470 printtab(t)"[CTRL 5][CTRL 9][SPACE
    ]3[SPACE]=>[SPACE]paars[8xSPACE]"
480 printtab(t)"[CTRL 6][CTRL 9][SPACE
    ]4[SPACE]=>[SPACE]groen[8xSPACE]"
490 printtab(t)"[CTRL 7][CTRL 9][SPACE
    ]5[SPACE]=>[SPACE]d.blauw[6xSPACE]
    "
500 printtab(t)"[CTRL 8][CTRL 9][SPACE
    ]6[SPACE]=>[SPACE]geel[9xSPACE]"
510 printtab(t)"[COM 3][CTRL 9][SPACE]
    7[SPACE]=>[SPACE]rose[9xSPACE]"
520 printtab(t)"[COM 6][CTRL 9][SPACE]
    8[SPACE]=>[SPACE]l.groen[6xSPACE]"
530 print
540 printtab(t)"[COM 5][CTRL 9][SPACE]
    ←[SPACE]=>[SPACE]nieuwe[SPACE]invo
    er"
550 print
560 printtab(t)"[COM 8][CTRL 9][SPACE]
    w[SPACE]=>[SPACE]witte[SPACE]pion[
    3xSPACE]"
570 printtab(t)"[COM 4][CTRL 9][SPACE]
    z[SPACE]=>[SPACE]zwarte[SPACE]pion
    [2xSPACE]"
580 return
590 print"[HOME]";:forgg=1to20:print"[
    CRSR-DOWN]";:nextgg:print"[4xCRSR-
    RIGHT]";
600 fort=1to30:print"[CTRL 9]"tt$;:for
    i=1to35:nexti
610 print"[4xCRSR-LEFT][CTRL 0]"tt$"[4
    xCRSR-LEFT]";:fori=1to15:nexti
620 pokes+3,t/2:nexttt:return
629 rem meer dan 13 beurten
630 gosub590
640 printsc$"[2xCRSR-DOWN]":gosub690
650 print"[HOME][6xCRSR-DOWN][2xCRSR-R
    IGHT][COM 8]helaas,[SPACE]maar[SPA
    CE]meer[SPACE]als[SPACE]13[SPACE]b
    eurten[SPACE]kan"
660 print"[CRSR-DOWN][2xCRSR-RIGHT]ik[
    SPACE]je[SPACE]toch[SPACE]echt[SPA
    CE]niet[SPACE]geven."
670 print"[CRSR-DOWN][2xCRSR-RIGHT]pro
    beer[SPACE]het[SPACE]nog[SPACE]een
    s[SPACE]!!"
680 goto390
689 rem omlijsten
690 print"[CTRL 7][CTRL 0][SPACE]U[36x
    SHIFT *]I"
700 forgg=1to5:print"[CRSR-RIGHT][SHIF
    T -][36xSPACE][SHIFT -]":next
710 print"[SPACE]J[36xSHIFT *]K"
720 return

730 rem geluid
740 pokes+5,0:pokes+6,248
750 pokes+1,68:pokes,149:pokes+4,17
760 pokes+4,16:return
770 pokes+5,0:pokes+6,253
780 pokes+1,8:pokes,147:pokes+4,65:ret
    urn
790 rem moeilijkheid
800 printsc$"[CRSR-DOWN]":gosub690:pri
    nt"[HOME][5xCRSR-DOWN][2xCRSR-RIGH
    T][COM 8]druk[SPACE]'f3'[SPACE]voo
    r[SPACE]moeilijkheid":ll=0
810 print"[CRSR-DOWN][2xCRSR-RIGHT]dru
    k[SPACE]'f1'[SPACE]om[SPACE]te[SPA
    CE]beginnen[CRSR-DOWN]"
820 le$(0)="[COM 7][CTRL 9]beginner[CT
    RL 0][SPACE]":le$(1)="[COM 7][CTRL
     9]gevorderd[CTRL 0]"
830 te$="[9xCRSR-LEFT]"
840 print"[HOME][9xCRSR-DOWN][2xCRSR-R
    IGHT]"le$(-ll)te$;:poke198,0
850 getg$:ifg$="[F1]"thenreturn
860 ifg$<>"[F3]"then850
870 ll=not(ll)
880 printle$(-ll)te$;:goto850
889 rem geen twee dezelfde
890 fory=0toa-1:ifb$(a)=b$(y)thenb$(a)
    ="":a=a-1:goto150
900 nexty:goto140
910 forgg=1to27:print"[COM I]";:next:r
    eturn
920 rem scherm mastermind
930 printsc$"[2xCRSR-DOWN][CTRL 6]"
940 print"[3xCRSR-RIGHT][COM A][4xSHIF
    T *][2xCOM R][4xSHIFT *][COM S]"
950 fort=1to13:t$=str$(t)
960 iflen(t$)<>3thent$=t$+"[SPACE]"
970 print t$"[SHIFT -][4xSPACE][2xSHIF
    T -][4xSPACE][SHIFT -]"
980 next
990 print"[3xSPACE][COM Q][4xSHIFT *][
    2xSHIFT +][4xSHIFT *][COM W]"
1000 print"[3xSPACE][SHIFT -][4xCOM +][
    2xSHIFT -]←←←←[SHIFT -]"
1010 print"[3xSPACE][COM Z][4xSHIFT *][
    2xCOM E][4xSHIFT *][COM X]"
1020 return
```

** EINDE LISTING mastermind

REGEL 10	160	REGEL 210	175
REGEL 20	15	REGEL 220	128
REGEL 30	203	REGEL 230	156
REGEL 40	168	REGEL 240	200
REGEL 50	0	REGEL 250	50
REGEL 60	250	REGEL 260	189
REGEL 70	115	REGEL 270	126
REGEL 80	14	REGEL 280	154
REGEL 90	138	REGEL 290	198
REGEL 100	18	REGEL 300	197
REGEL 110	94	REGEL 310	5
REGEL 120	44	REGEL 320	52
REGEL 129	74	REGEL 330	246
REGEL 130	111	REGEL 340	33
REGEL 140	3	REGEL 349	31
REGEL 150	195	REGEL 350	192
REGEL 160	22	REGEL 360	253
REGEL 169	179	REGEL 370	125
REGEL 170	173	REGEL 380	194
REGEL 180	144	REGEL 390	219
REGEL 190	51	REGEL 400	50
REGEL 200	196	REGEL 410	88

```
REGEL 420   154
REGEL 430   138
REGEL 439   103
REGEL 440   176
REGEL 450   11
REGEL 460   199
REGEL 470   208
REGEL 480   87
REGEL 490   203
REGEL 500   123
REGEL 510   144
REGEL 520   80
REGEL 530   153
REGEL 540   33
REGEL 550   153
REGEL 560   63
REGEL 570   142
REGEL 580   142
REGEL 590   208
REGEL 600   231
REGEL 610   152
REGEL 620   254
REGEL 629   4
REGEL 630   43
REGEL 640   31
REGEL 650   130
REGEL 660   112
REGEL 670   105
REGEL 680   37
REGEL 689   68
REGEL 690   44
REGEL 700   117
REGEL 710   114
REGEL 720   142
REGEL 730   73
REGEL 740   243
REGEL 750   227
REGEL 760   35
REGEL 770   239
REGEL 780   118
REGEL 790   253
REGEL 800   199
REGEL 810   175
REGEL 820   251
REGEL 830   56
REGEL 840   211
REGEL 850   236
REGEL 860   104
REGEL 870   219
REGEL 880   58
REGEL 889   38
REGEL 890   52
REGEL 900   51
REGEL 910   119
REGEL 920   69
REGEL 930   215
REGEL 940   246
REGEL 950   141
REGEL 960   229
REGEL 970   201
REGEL 980   130
REGEL 990   241
REGEL 1000   101
REGEL 1010   169
REGEL 1020   142

READY.
```

Snake

Tommy van den Eynde, ja ik kan het ook niet helpen, weer uit Belgie maakte het volgende programma. Snake, slangen dus. Het is een spel voor twee personen. Er verschijnen twee slangen op het scherm die steeds langer worden. Er mag nergens tegen aan gebotst worden. De bedoeling is de tegenstander in te sluiten zo dat deze gedwongen ergens tegen op botst.

```
5 rem *snakes : door tommy van den e
  ynde*
7 rem *geraardsbergen (belgie)*
10 g=0:bb=0:print"[SHIFT CLR]":goto10
   00
45 poke53280,0:poke53281,0:a1=1513:a2
   =1534:z1=1:z2=-1:b1=0:b2=0
60 pokea1,81:pokea2,81:pokea1+54272,2
   :pokea2+54272,6
70 fori=1024to1063:pokei,160:pokei+96
   0,160:nexti
80 fori=1064to1944step40:pokei,160:po
   kei+39,160:next:ifbb=0thengoto95
92 poke1043,42:poke1044,42:poke2003,4
   2:poke2004,42:poke1344,42:poke1664
   ,42
94 poke1383,42:poke1703,42
95 forj=1to10:e=1029:pokee,18:pokee+1
   ,5:pokee+2,1:pokee+3,4:pokee+4,25
97 forabc=1to100:nextabc
98 fori=1029to1033:pokei,160:nexti:ne
   xtj
100 y1=peek(56321):ify1=254thenz1=-40
120 ify1=253thenz1=40
130 ify1=251thenz1=-1
140 ify1=247thenz1=1
145 ify1=255thenz1=z1
150 y2=peek(56320):ify2=126thenz2=-40
170 ify2=125thenz2=40
180 ify2=123thenz2=-1
190 ify2=119thenz2=1
192 ify2=127thenz2=z2
195 ifpeek(a1+z1)<>32andpeek(a1+z1)<>4
    2thenb1=1
196 ifpeek(a2+z2)<>32andpeek(a2+z2)<>4
    2thenb2=1
197 ifb1=1orb2=1thengosub2010
198 ifb1=1orb2=1thengoto290
200 ifpeek(a1+z1)=42thengoto1260
202 ifpeek(a2+z2)=42thengoto1280
205 pokea1+z1+54272,2:pokea1+54272,2:p
    okea1+z1,81:pokea1,160:a1=a1+z1
210 pokea2+z2+54272,6:pokea2+54272,6:p
    okea2+z2,81:pokea2,160:a2=a2+z2:go
    to100
290 fori=1to1000:next:print"[SHIFT CLR
    ]"
300 ifb1=1andb2=1thenprint:printtab(14
    )"no[SPACE]winner":goto325
310 ifb1=1thenprint:printtab(12)"winne
    r[SPACE]:[SPACE]player[SPACE]2":go
    to325
320 ifb2=1thenprint:printtab(12)"winne
    r[SPACE]:[SPACE]player[SPACE]1"
325 fori=1to1000:nexti:poke214,8:print
    :poke211,5:print"press[SPACE]fire[
    SPACE]button[SPACE]";
332 print"to[SPACE]play[SPACE]again":p
    rint:printtab(15)"'q'[SPACE]to[SPA
    CE]quit"
335 geta$:ifa$="q"thenprint"[SHIFT CLR
    ]":poke774,0:new:end
340 p=peek(56321):ifp=239thengoto370
360 goto335
370 fori=1to200:nexti:goto10
1000 print"[CTRL 2]":poke53280,0:poke53
     281,9:poke214,4:print:poke211,8:pr
     int"s[SPACE]n[SPACE]a";
1010 print"[SPACE]k[SPACE]e[SPACE]s[3xS
     PACE](2[SPACE]players)":poke214,11
     :print:poke211,17:print"game";g
1030 poke214,20:print:poke211,8:print"u
     se[SPACE]joystick[SPACE]1[SPACE]to
     [SPACE]select"
1040 g2=peek(56321):ifg2=251org2=253the
     ng=g-1
1050 ifg2=254org2=247theng=g+1
1060 ifg=-1theng=9
1070 ifg=10theng=0
1080 ifg2=239thenprint"[SHIFT CLR]":got
     o1100
1090 goto1000
1100 ong+1goto45,1110,1120,1140,1160,11
     90,1210,1250,1320,1325
1110 fori=1403to1643step40:pokei,160:po
     kei+1,160:nexti:goto45
```

```
1120 fori=1359to1687step41:pokei,160:po
     kei+1,160:nexti
1130 fori=1367to1679step39:pokei,160:po
     kei+1,160:nexti:goto45
1140 fori=1305to1323step2:pokei,160:pok
     ei+400,160:pokei+19,160:pokei+419,
     160
1150 nexti:goto45
1160 fori=1307to1319step2:pokei,160:pok
     ei+21,160:pokei+400,160:pokei+421,
     160
1170 nexti:fori=1346to1666step80:pokei,
     160:pokei+14,160:pokei+21,160
1180 pokei+35,160:nexti:goto45
1190 fori=1267to1747step40:pokei,160:po
     kei+12,160:pokei+21,160:pokei+33,1
     60
1200 nexti:goto45
1210 fori=1313to1316:pokei,160:pokei+18
     ,160:pokei+400,160:pokei+418,160:n
     exti
1220 fori=1439to1599step40:pokei,160:po
     kei+9,160:nexti
1230 fori=1357to1398step41:pokei,160:po
     kei+292,160:nexti
1240 fori=1370to1409step39:pokei,160:po
     kei+268,160:nexti:goto45
1250 bb=1:goto45
1260 pokea1,160:ifz1=-1thena1=a1+37:got
     o205
1265 ifz1=1thena1=a1-37:goto205
1270 ifz1=-40thena1=a1+880:goto205
1275 ifz1=40thena1=a1-880:goto205
1280 pokea2,160:ifz2=-1thena2=a2+37:got
     o205
1290 ifz2=1thena2=a2-37:goto205
1300 ifz2=-40thena2=a2+880:goto205
1310 ifz2=40thena2=a2-880:goto205
1320 d=40:goto1328
1325 d=100:goto1328
1328 fori=1tod:pt=int(918*rnd(1)):poke1
     065+pt,160:nexti
1360 fori=1513to1534:pokei,32:nexti:got
     o45
2010 v1=54272:v2=54279:v3=54286:l=54293
     :h=54294:ru=54295:pl=54296
2080 pokev1+4,0:pokev2+4,0:pokev3+4,0:a
     =0:d=10:s=8:r=11:l=1000
2110 pokeru,0:pokepl,15:pokev1,0:pokev1
     +1,5:pokev3,0:pokev3+1,10
2150 pokev1+5,16*a+d:pokev1+6,16*s+r:po
     kev1+4,129:pokev3+4,23:forj=0tol:n
     ext
2180 pokev1+4,128:pokev3+4,16:return

     ** EINDE LISTING snake
```

```
REGEL 5      221
REGEL 7      216
REGEL 10     247
REGEL 45     148
REGEL 60     26
REGEL 70     185
REGEL 80     31
REGEL 92     209
REGEL 94     38
REGEL 95     214
REGEL 97     225
REGEL 98     155
REGEL 100    223
REGEL 120    169
REGEL 130    31
REGEL 140    121
REGEL 145    210
REGEL 150    223
REGEL 170    169
REGEL 180    31
REGEL 190    121
REGEL 192    211
REGEL 195    62
REGEL 196    67
REGEL 197    223
REGEL 198    179
REGEL 200    86
REGEL 202    90
REGEL 205    177
REGEL 210    22
REGEL 290    120
REGEL 300    60
REGEL 310    208
REGEL 320    115
REGEL 325    93
REGEL 332    183
REGEL 335    243
REGEL 340    69
REGEL 360    36
REGEL 370    12
REGEL 1000   226
REGEL 1010   216
REGEL 1030   96
REGEL 1040   255
REGEL 1050   139
REGEL 1060   57
REGEL 1070   181
REGEL 1080   240
REGEL 1090   74
REGEL 1100   46
REGEL 1110   137
REGEL 1120   112
REGEL 1130   163
REGEL 1140   155
REGEL 1150   247
REGEL 1160   148
REGEL 1170   138
REGEL 1180   230
REGEL 1190   110
REGEL 1200   247
REGEL 1210   196
REGEL 1220   120
REGEL 1230   217
REGEL 1240   3
REGEL 1250   147
REGEL 1260   85
REGEL 1265   165
REGEL 1270   184
REGEL 1275   14
REGEL 1280   89
REGEL 1290   168
REGEL 1300   187
REGEL 1310   17
REGEL 1320   235
REGEL 1325   24
REGEL 1328   105
REGEL 1360   147
REGEL 2010   227
REGEL 2080   122
REGEL 2110   89
REGEL 2150   136
REGEL 2180   86

READY.
```

Analoge klok C-16

Niek Meyer moet gedacht hebben, blijven jullie wat meer bij de tijd en bedacht een analoge en digitale klok ineen. Een leuk programma, we zullen de tijd nooit meer vergeten.

```
 2 trap102
 4 scnclr
 6 char,7,10,"":input"de[SPACE]huidig
   e[SPACE]tijd[SPACE]is?[SPACE]00000
   0[8xCRSR-LEFT]";ti$
 8 graphic1,1
10 x=159:y=99:draw,x,y
12 box,39,6,279,192
14 box,43,11,275,187
16 box,47,16,271,182
18 draw,43,9to275,9
20 draw,43,189to275,189
22 draw,47,13to271,13
24 draw,47,185to271,185
26 char,16,1,"[SPACE]cbm[SPACE]16[SPA
   CE]"
28 char,15,23,"[SPACE]00:00:00[SPACE]
   "
30 char,24,5,"1":char,27,8,"2":char,2
   8,12,"3":char,27,16,"4"
32 char,24,19,"5":char,19,21,"6":char
   ,14,19,"7":char,11,16,"8"
34 char,10,12,"9":char,11,8,"10":char
```

```
    ,14,5,"11":char,19,4,"12"
 36 s$=right$(ti$,2):s=val(s$)
 38 m$=mid$(ti$,3,2):m=val(m$)
 40 u$=left$(ti$,2):u=val(u$):gosub86
 42 char,16,23,u$:char,19,23,m$:char,2
    2,23,s$
 44 do
 46 dountilt$<>ti$
 48 circle,x,y,0,55,270,0,(6*m),120
 50 circle,x,y,0,35,270,0,(30*u),120
 52 loop:t$=ti$
 54 circle0,x,y,0,60,270,0,(6*s),120
 56 s$=right$(ti$,2):s=val(s$)
 58 vol5:sound1,810,4
 60 circle,x,y,0,60,270,0,(6*s),120
 62 char,22,23,s$
 64 ifs<>0thenloop
 66 circle0,x,y,0,55,270,0,(6*m),120
 68 m$=mid$(ti$,3,2):m=val(m$)
 70 char,19,23,m$
 72 circle,x,y,0,55,270,0,(6*m),120
 74 circle0,x,y,0,35,270,0,(30*u),120
 76 u$=left$(ti$,2):u=val(u$):gosub86
 78 char,16,23,u$
 80 circle,x,y,0,35,270,0,(30*u),120
 82 loop
 84 :
 86 ifm>10andm<20thenu=u+.2
 88 ifm>20andm<30thenu=u+.4
 90 ifm>30andm<40thenu=u+.6
 92 ifm>40andm<50thenu=u+.8
 94 ifm>50thenu=u+1
 96 ifu>12thenu=u-12
 98 return
100 :
102 graphic0:printel;err$(er):end
```

REGEL 2	106	REGEL 62	164
REGEL 4	232	REGEL 64	5
REGEL 6	213	REGEL 66	153
REGEL 8	108	REGEL 68	124
REGEL 10	136	REGEL 70	164
REGEL 12	113	REGEL 72	105
REGEL 14	152	REGEL 74	204
REGEL 16	152	REGEL 76	104
REGEL 18	132	REGEL 78	169
REGEL 20	86	REGEL 80	156
REGEL 22	218	REGEL 82	236
REGEL 24	78	REGEL 84	58
REGEL 26	121	REGEL 86	8
REGEL 28	7	REGEL 88	12
REGEL 30	241	REGEL 90	16
REGEL 32	93	REGEL 92	20
REGEL 34	67	REGEL 94	204
REGEL 36	46	REGEL 96	5
REGEL 38	124	REGEL 98	142
REGEL 40	104	REGEL 100	58
REGEL 42	101	REGEL 102	34
REGEL 44	235	REGEL 990	40
REGEL 46	132	REGEL 1010	102
REGEL 48	105	REGEL 1020	100
REGEL 50	156	REGEL 1030	187
REGEL 52	17	REGEL 1040	249
REGEL 54	155	REGEL 1050	156
REGEL 56	46	REGEL 1060	248
REGEL 58	122	REGEL 1070	40
REGEL 60	107	REGEL 1080	38

REGEL 1100	104	REGEL 1300	38
REGEL 1110	136	REGEL 1320	205
REGEL 1120	245	REGEL 1330	94
REGEL 1170	54	REGEL 1340	125
REGEL 1180	82	REGEL 1350	39
REGEL 1190	81	REGEL 1355	3
REGEL 1200	12	REGEL 1360	188
REGEL 1210	53	REGEL 1370	145
REGEL 1220	189	REGEL 1380	177
REGEL 1230	217	REGEL 1390	189
REGEL 1240	247	REGEL 1400	0
REGEL 1250	187	REGEL 1410	230
REGEL 1260	172		
REGEL 1270	213		
REGEL 1280	5	READY.	
REGEL 1290	132		

Etiketten 128

Heeft u de beschikking over een commodore 128, een tachtig koloms monitor en een printer, dan is het een koud kunstje om met dit programma etiketten af te drukken. Alleen invullen wat er op het etiket moet komen en de rest wordt voor u gedaan.

```
 10 rem ************************
 20 rem ** printer star ln-10   **
 30 rem ** 5 regels per etiket **
 40 rem ** commodore 128        **
 50 rem ** line feed van        **
 60 rem ** printer moet aan !   **
 70 rem ************************
 80 print"[SHIFT CLR]":fori=1to500:nex
    t
 90 printtab(28)"[21xCRSR-DOWN][COM 6]
    gemaakt[SPACE]by[SPACE]mr.[SPACE]b
    art[SPACE]2001"
100 printtab(28)"[COM 7][24xCOM T]
110 print"[30xCRSR-UP]"
120 printtab(33)"[COM 7]etiketten[SPAC
    E]128"
130 printtab(33)"[COM 6][13xCOM T]"
140 fori=1to500:next
150 print"[COM 7][2xCRSR-DOWN][4xCRSR-
    RIGHT]etiketten[SPACE]128[2xSPACE]
    (monitor[SPACE]op[SPACE]80[SPACE]t
    ekens)
160 fori=1to3000:next
170 print"[SHIFT CLR]"
180 printtab(28)"[21xCRSR-DOWN][COM 3]
    gemaakt[SPACE]by[SPACE]mr.[SPACE]b
    art[SPACE]2001"
190 printtab(28)"[CTRL 8][24xCOM T]
200 print"[30xCRSR-UP]"
210 printtab(33)"[CTRL 8]etiketten[SPA
    CE]128"
220 printtab(33)"[COM 3][13xCOM T]"
230 fori=1to500:next
240 print"[2xCRSR-DOWN][4xCRSR-RIGHT][
    COM 3]wat[SPACE]wilt[SPACE]u[SPACE
    ]op[SPACE]u[SPACE]etiket[SPACE]heb
    ben[SPACE][CTRL 8]?"
250 input"[2xCRSR-DOWN][4xCRSR-RIGHT]"
    ;a$
```

```
260 input"[4xCRSR-RIGHT]";b$
270 input"[4xCRSR-RIGHT]";c$
280 input"[4xCRSR-RIGHT]";d$
290 input"[4xCRSR-RIGHT]";e$
300 open4,4:cmd4
310 print a$
320 print b$
330 print c$
340 print d$
350 print e$
355 printchr$(10)
360 print#4:close4
370 print"[SHIFT CLR]":fori=1to500:nex
    t
380 printtab(28)"[21xCRSR-DOWN][CTRL 5
    ]gemaakt[SPACE]by[SPACE]mr.[SPACE]
    bart[SPACE]2001"
390 printtab(28)"[COM 4][24xCOM T]
400 print"[30xCRSR-UP]"
410 input"[CRSR-DOWN][4xCRSR-RIGHT][CO
    M 4]wilt[SPACE]u[SPACE]nog[SPACE]m
    eer[SPACE]etiketten[SPACE]hebben[S
    PACE][CTRL 5]";j$
420 ifj$="j"then170
430 print"[SHIFT CLR]":end

    ** EINDE LISTING etiketten 128
```

REGEL 10	169	REGEL 250	255
REGEL 20	189	REGEL 260	222
REGEL 30	219	REGEL 270	223
REGEL 40	119	REGEL 280	224
REGEL 50	88	REGEL 290	225
REGEL 60	129	REGEL 300	62
REGEL 70	169	REGEL 310	254
REGEL 80	76	REGEL 320	255
REGEL 90	95	REGEL 330	0
REGEL 100	211	REGEL 340	1
REGEL 110	219	REGEL 350	2
REGEL 120	241	REGEL 355	18
REGEL 130	239	REGEL 360	218
REGEL 140	162	REGEL 370	76
REGEL 150	125	REGEL 380	98
REGEL 160	208	REGEL 390	208
REGEL 170	112	REGEL 400	219
REGEL 180	92	REGEL 410	29
REGEL 190	215	REGEL 420	120
REGEL 200	219	REGEL 430	42
REGEL 210	245		
REGEL 220	236		
REGEL 230	162		
REGEL 240	197		

READY.

Adresbak 128

Je hoeft nooit meer te zoeken naar een adres van familie of kennisen, zelf een hele verenigings adressen administratie is er in op te bergen. Het enige wat u hiervoor moet doen is de listing van Paul Siem in te tikken. Het programma is geheel menu gestuurd en heeft daarom geen verdere uitleg nodig.

```
  1 rem  adresbak 128 / commodore-128
  2 rem  door paul siem
  3 rem  uit rozendaal
  4 rem
 10 n=0:n=f:vol15:dima$(400,7)
 20 forq=1to7
 30 readt$(q):nextq
 40 data"welke[SPACE]achternaam","welk
    e[SPACE]voornaam","welk[SPACE]adre
    s","welke[SPACE]postcode","welke[S
    PACE]plaats","welk[SPACE]tel.numme
    r","welke[SPACE]diversen"
 50 trap2070:scnclr:print"[CRSR-DOWN]"
    :color5,14
 60 printtab(13)"***********"
 70 printtab(13)"*[CTRL 9][9xSPACE][CT
    RL 0]*"
 80 printtab(13)"*[CTRL 9]hoofdmenu[CT
    RL 0]*"
 90 printtab(13)"*[CTRL 9][9xSPACE][CT
    RL 0]*"
100 printtab(13)"***********"
110 print"[CRSR-DOWN]"
120 printtab(4)"[CTRL 9][SPACE]1[SPACE
    ][CTRL 0][SPACE]=[SPACE]gegevens[S
    PACE]laden"
130 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]2[SPACE][CTRL 0][SPACE]=[SPACE
    ]gegevens[SPACE]invoeren"
140 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]3[SPACE][CTRL 0][SPACE]=[SPACE
    ]gegevens[SPACE]zoeken[SPACE]/[SPA
    CE]muteren"
150 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]4[SPACE][CTRL 0][SPACE]=[SPACE
    ]gegevens[SPACE]sorteren"
160 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]5[SPACE][CTRL 0][SPACE]=[SPACE
    ]gegevens[SPACE]naar[SPACE]de[SPAC
    E]printer"
170 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]6[SPACE][CTRL 0][SPACE]=[SPACE
    ]gegevens[SPACE]saven"
180 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]7[SPACE][CTRL 0][SPACE]=[SPACE
    ]directory"
190 printtab(4)"[CRSR-DOWN][CTRL 9][SP
    ACE]8[SPACE][CTRL 0][SPACE]=[SPACE
    ]stoppen"
200 getkeyw$
210 w=val(w$)
220 ifw=>1andw<=8then230:elsegosub1210
    :goto200
230 onwgoto350,410,610,860,970,1070,30
    0,240
```

```
240 gosub1140
250 printtab(4)"[CTRL 9][SPACE]1[SPACE
    ][CTRL 0][SPACE]=[SPACE]stoppen"
260 getkeyw$
270 ifw$="←"then50
280 ifw$="1"then290:elsegosub1210:goto
    260
290 scnclr:end
300 scnclr:print"[2xCRSR-DOWN]"
310 sound4,2,39,24
320 directory
330 print"[2xCRSR-DOWN]druk[SPACE]een[
    SPACE]toets[SPACE]voor[SPACE]menu"
    :sound0,0,39,24:getkeyw$
340 goto50
350 gosub1140
360 printtab(4)"[CTRL 9][SPACE]1[SPACE
    ][CTRL 0][SPACE]=[SPACE]gegevens[S
    PACE]laden[SPACE]van[SPACE]schi'´"
370 getkeyw$
380 ifw$="1"thengosub2190:goto400
390 ifw$="←"then50:elsegosub1210:goto3
    70
400 goto50
410 n=n+1
420 scnclr:print"[CRSR-DOWN]":fo=0
430 printtab(2)"[COM 6]achternaam[SPAC
    E][CTRL 4]";:inputa$(n,1)
440 printtab(2)"[CRSR-DOWN][COM 6]voor
    naam[3xSPACE][CTRL 4]";:inputa$(n,
    2)
450 printtab(2)"[CRSR-DOWN][COM 6]adre
    s[6xSPACE][CTRL 4]";:inputa$(n,3)
460 printtab(2)"[CRSR-DOWN][COM 6]post
    code[3xSPACE][CTRL 4]";:inputa$(n,
    4)
470 printtab(2)"[CRSR-DOWN][COM 6]plaa
    ts[5xSPACE][CTRL 4]";:inputa$(n,5)
480 printtab(2)"[CRSR-DOWN][COM 6]tel.
    nummer[SPACE][CTRL 4]";:inputa$(n,
    6)
490 printtab(2)"[CRSR-DOWN][COM 6]dive
    rsen[3xSPACE][CTRL 4]";:inputa$(n,
    7)
500 print"[2xCRSR-DOWN]"
510 printtab(2)"[COM 6]is[SPACE]dit[SP
    ACE]correct[SPACE](j/n)[SPACE][CTR
    L 4]";
520 getkeyw$
530 ifw$="n"thenfo=1:gosub1230:goto550
540 ifw$<>"j"thengosub1210:goto520
550 printw$
560 printtab(2)"[CRSR-DOWN][COM 6]nog[
    SPACE]een[SPACE]keer[SPACE](j/n)[S
    PACE][CTRL 4]";
570 getkeyw$
580 ifw$="n"thenprintw$:goto50
590 ifw$="j"then600:else:gosub1210:got
    o570
600 printw$:iffo=1then420:else410
610 gosub1140
620 printtab(4)"waarop[SPACE]wilt[SPAC
    E]u[SPACE]zoeken[SPACE]/[SPACE]mut
    eren[SPACE]?"
630 gosub1730
640 getkeyw$
650 k=val(w$)
660 ifw$="←"then50
670 ifk=>1andk<=7then680:elsegosub1210
    :goto640
680 gosub1140
690 printtab(4)t$(k);"[SPACE]wilt[SPAC
    E]u[SPACE]zoeken"
700 printtab(4)"[CTRL 4][CRSR-DOWN]";
710 inputan$
720 ifan$="←"thenprint"[COM 6]":goto61
    0
730 r=4:gosub1240
740 getkeyw$
750 ifw$="←"then680
760 ifw$="v"then810
770 ifw$="t"then820
780 ifw$="*"then830
790 ifw$="@"then840
800 ifw$="p"then850:elsegosub1210:goto
    740
810 r=2:gosub1240:goto740
820 gosub1330:goto740
830 gosub1550:goto740
840 gosub1640:goto740
850 gosub1950:goto740
860 gosub1140
870 printtab(4)"waarop[SPACE]wilt[SPAC
    E]u[SPACE]sorteren[SPACE]?"
880 gosub1730
890 getkeyw$
900 k=val(w$)
910 ifw$="←"then50
920 ifk=>1andk<=7then940
930 gosub1210:goto890
940 scnclr
950 printtab(2)"[2xCRSR-DOWN]een[SPACE
    ]momentje[SPACE]a.u.b..."
960 gosub1810:gosub1210:printtab(2)"[2
    xCRSR-DOWN][COM 6][SPACE]gesorteer
    d[SPACE]":goto50
970 gosub1140
980 printtab(4)"waarop[SPACE]wilt[SPAC
    E]u[SPACE]uitprinten[SPACE]?"
990 gosub1730
1000 getkeyw$:ifw$="←"then50
1010 k=val(w$):ifk=>1andk<=7then1020:el
     segosub1210:goto1000
1020 gosub1140
1030 printtab(4)t$(k)"[SPACE]wilt[SPACE
     ]u[SPACE]uitprinten"
1040 printtab(4)"[CTRL 4][CRSR-DOWN]";:
     inputam$
1050 ifam$="←"then970
1060 gosub2420:goto50
1070 gosub1140
1080 printtab(4)"[CTRL 9][SPACE]1[SPACE
     ][CTRL 0][SPACE]=[SPACE]gegevens[S
     PACE]saven[SPACE]op[SPACE]schijf"
1090 getkeyw$
1100 ifw$="←"then50
1110 ifw$="1"then1130
1120 gosub1210:goto1090
1130 ifn=0thengosub1210:goto50:else:gos
     ub2310:goto50
1140 scnclr:print"[COM 6][CRSR-DOWN]"
1150 printtab(11)"*****************"
1160 printtab(11)"*[CTRL 9][15xSPACE][C
     TRL 0]*"
1170 printtab(11)"*[CTRL 9]typ[SPACE]←[
     SPACE]voor[SPACE]menu[CTRL 0]*"
1180 printtab(11)"*[CTRL 9][15xSPACE][C
     TRL 0]*"
```

```
1190 printtab(11)"*****************"
1200 print"[CRSR-DOWN]":return
1210 pen"c"
1220 return
1230 fory=1to7:a$(n,y)="":nexty:return
1240 ifr=4then1260
1250 f=f+1:goto1270
1260 f=0
1270 do:geti$
1280 ifan$=a$(f,k)thengosub1400:return
1290 ifi$="←"thenprinttab(r)"[CRSR-DOWN
     ][COM 6][CTRL 9][SPACE]onderbrekin
     g[SPACE]":gosub1210:goto1320
1300 f=f+1:loopwhilef<400:gosub1210
1310 printtab(r)"[CRSR-DOWN][COM 6][CTR
     L 9][SPACE]geen[SPACE]gegevens[SPA
     CE]in[SPACE]het[SPACE]bestand[SPAC
     E]"
1320 return
1330 iff=0then1380
1340 do:f=f-1
1350 ifan$=a$(f,k)thengosub1400:goto137
     0
1360 loopuntilf=0:goto1380
1370 return
1380 printtab(2)"[CRSR-DOWN][COM 6][CTR
     L 9][SPACE]geen[SPACE]gegevens[SPA
     CE]in[SPACE]het[SPACE]bestand[SPAC
     E]"
1390 gosub1210:return
1400 scnclr
1410 printtab(2)"[CRSR-DOWN][COM 6]acht
     ernaam[SPACE]:[2xSPACE][CTRL 4]";a
     $(f,1)
1420 printtab(2)"[CRSR-DOWN][COM 6]voor
     naam[3xSPACE]:[2xSPACE][CTRL 4]";a
     $(f,2)
1430 printtab(2)"[CRSR-DOWN][COM 6]adre
     s[6xSPACE]:[2xSPACE][CTRL 4]";a$(f
     ,3)
1440 printtab(2)"[CRSR-DOWN][COM 6]post
     code[3xSPACE]:[2xSPACE][CTRL 4]";a
     $(f,4)
1450 printtab(2)"[CRSR-DOWN][COM 6]plaa
     ts[5xSPACE]:[2xSPACE][CTRL 4]";a$(
     f,5)
1460 printtab(2)"[CRSR-DOWN][COM 6]tel.
     nummer[SPACE]:[2xSPACE][CTRL 4]";a
     $(f,6)
1470 printtab(2)"[CRSR-DOWN][COM 6]dive
     rsen[3xSPACE]:[2xSPACE][CTRL 4]";a
     $(f,7)
1480 printtab(2)"[COM 6][2xCRSR-DOWN]ty
     p[SPACE]'←'[SPACE]voor[SPACE]menu"
1490 printtab(2)"typ[SPACE]'v'[SPACE]vo
     or[SPACE]verder"
1500 printtab(2)"typ[SPACE]'t'[SPACE]vo
     or[SPACE]terug"
1510 printtab(2)"typ[SPACE]'*'[SPACE]vo
     or[SPACE]veranderen"
1520 printtab(2)"typ[SPACE]'@'[SPACE]vo
     or[SPACE]wissen"
1530 printtab(2)"typ[SPACE]'p'[SPACE]vo
     or[SPACE]printen"
1540 return
1550 gosub1210:printtab(2)"[CRSR-DOWN]w
     at[SPACE]wilt[SPACE]u[SPACE]verand
     eren"
1560 print"[23xCRSR-UP][14xCRSR-RIGHT]"
     ;
1570 fory=1to7
1580 inputa$(f,y)
1590 print"[CRSR-DOWN][14xCRSR-RIGHT]";
1600 nexty
1610 print"[8xCRSR-DOWN][14xCRSR-LEFT][
     25xSPACE]"
1620 gosub1210
1630 return
1640 gosub1210:printtab(2)"[CRSR-DOWN][
     COM 6]zeker[SPACE]weten[SPACE](j/n
     )[SPACE][CTRL 4]"
1650 getkeyw$:ifw$="n"then1710
1660 ifw$="j"then1670:elsegosub1210:got
     o1650
1670 fory=1to7
1680 a$(f,y)=""
1690 nexty
1700 gosub1210
1710 print"[CRSR-UP][23xSPACE][COM 6][2
     xCRSR-UP]"
1720 return
1730 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]1[SPACE][CTRL 0][SPACE]=[SPACE
     ]achternaam"
1740 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]2[SPACE][CTRL 0][SPACE]=[SPACE
     ]voornaam"
1750 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]3[SPACE][CTRL 0][SPACE]=[SPACE
     ]adres"
1760 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]4[SPACE][CTRL 0][SPACE]=[SPACE
     ]postcode"
1770 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]5[SPACE][CTRL 0][SPACE]=[SPACE
     ]plaats"
1780 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]6[SPACE][CTRL 0][SPACE]=[SPACE
     ]tel.nummer"
1790 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]7[SPACE][CTRL 0][SPACE]=[SPACE
     ]diversen"
1800 return
1810 forx=1ton-1
1820 forz=1ton-x
1830 ifa$(z,k)=>a$(z+1,k)thengosub1860
1840 nextz:nextx
1850 return
1860 fory=1to7
1870 w$=a$(z,y):a$(z,y)=a$(z+1,y):a$(z+
     1,y)=w$
1880 nexty
1890 return
1900 open1,8,15
1910 input#1,ff,fb$,sp,se
1920 printtab(3)"[COM 6][2xCRSR-DOWN]";
     ff;fb$;sp;se
1930 close1
1940 return
1950 open4,4
1960 print#4:print#4
1970 print#4,"[2xSPACE]achternaam[SPACE
     ]:[SPACE]";a$(n,1)
1980 print#4,"[2xSPACE]voornaam[3xSPACE
     ]:[SPACE]";a$(n,2)
1990 print#4,"[2xSPACE]adres[6xSPACE]:[
     SPACE]";a$(n,3)
2000 print#4,"[2xSPACE]postcode[3xSPACE
```

```
     ]:[SPACE]";a$(n,4)
2010 print#4,"[2xSPACE]plaats[5xSPACE]:
     [SPACE]";a$(n,5)
2020 print#4,"[2xSPACE]tel.nummer[SPACE
     ]:[SPACE]";a$(n,6)
2030 print#4,"[2xSPACE]diversen[3xSPACE
     ]:[SPACE]";a$(n,7)
2040 close4
2050 gosub1210
2060 return
2070 scnclr:print"[3xCRSR-DOWN]":close2
     :trap2070
2080 gosub1210
2090 printtab(4)"[COM 6][CTRL 9]" err$(
     er);"[SPACE]error[SPACE]in";el;"[C
     TRL 0]"
2100 printtab(4)"[2xCRSR-DOWN]wilt[SPAC
     E]u"
2110 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]1[SPACE][CTRL 0][SPACE]=[SPACE
     ]gegevens[SPACE]wegschrijven"
2120 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]2[SPACE][CTRL 0][SPACE]=[SPACE
     ]stoppen"
2130 printtab(4)"[CRSR-DOWN][CTRL 9][SP
     ACE]←[SPACE][CTRL 0][SPACE]=[SPACE
     ]terug[SPACE]naar[SPACE]menu"
2140 getkeyw$
2150 ifw$="←"then50
2160 w=val(w$)
2170 onwgoto1070,240
2180 gosub1210:goto2140
2190 printtab(4);:input"[2xCRSR-DOWN]fi
     lenaam[SPACE][CTRL 4]";fi$
2200 iffi$="←"then50
2210 open2,8,2,fi$+",s,r"
2220 draw:x=0
2230 do:x=x+1:fory=1to7
2240 input#2,a$(x,y)
2250 nexty:n=n+1:loopwhilest=0
2260 close2:locate
2270 gosub1900
2280 ifff=0andfb$="ok"andsp=0andse=0the
     n2300
2290 printtab(4)"[2xCRSR-DOWN][COM 6][C
     TRL 9][SPACE]leesfout[SPACE]!![SPA
     CE]":printtab(4)"[CRSR-DOWN]probee
     r[SPACE]het[SPACE]nog[SPACE]eens..
     .":gosub1210:5:goto350
2300 printtab(4)"[CRSR-DOWN][COM 6][CTR
     L 9][SPACE]ok[SPACE]":2:return
2310 printtab(4);:input"[2xCRSR-DOWN]fi
     lenaam[SPACE][CTRL 4]";fi$
2320 iffi$="←"then50
2330 open2,8,2,"@0:"+fi$+",s,w"
2340 draw
2350 forx=1ton-1:fory=1to7
2360 print#2,a$(x,y)
2370 nexty:nextx
2380 close2:locate
2390 gosub1900
2400 ifff=0andfb$="ok"andsp=0andse=0the
     n2410
2410 printtab(4)"[CRSR-DOWN][COM 6][CTR
     L 9][SPACE]ok[SPACE]":2:return
2420 scnclr:printtab(2)"[COM 6][2xCRSR-
     DOWN]doe[SPACE]de[SPACE]printer[SP
     ACE]aan[SPACE]en[SPACE]druk[SPACE]
     een[SPACE]toets.":gosub1210
2430 printtab(2)"[CRSR-DOWN]hou[SPACE]t
     ijdens[SPACE]het[SPACE]printen[SPA
     CE][CTRL 9][SPACE]s[SPACE][CTRL 0]
     [SPACE]voor":printtab(2)"[CRSR-DOW
     N]stoppen[SPACE]aan.[SPACE]hij[SPA
     CE]print[SPACE]dan[SPACE]alleen[SP
     ACE]nog":printtab(2)"[CRSR-DOWN]he
     t[SPACE]laatste[SPACE]adres[SPACE]
     af.":getkeyw$
2440 fory=1ton
2450 ifa$(y,k)=am$then2480
2460 nexty
2470 return
2480 getr$:ifr$="s"then2590
2490 open4,4
2500 print#4:print#4
2510 print#4,"[2xSPACE]achternaam[SPACE
     ]:[SPACE]";a$(y,1)
2520 print#4,"[2xSPACE]voornaam[3xSPACE
     ]:[SPACE]";a$(y,2)
2530 print#4,"[2xSPACE]adres[6xSPACE]:[
     SPACE]";a$(y,3)
2540 print#4,"[2xSPACE]postcode[3xSPACE
     ]:[SPACE]";a$(y,4)
2550 print#4,"[2xSPACE]plaats[5xSPACE]:
     [SPACE]";a$(y,5)
2560 print#4,"[2xSPACE]tel.nummer[SPACE
     ]:[SPACE]";a$(y,6)
2570 print#4,"[2xSPACE]diversen[3xSPACE
     ]:[SPACE]";a$(y,7)
2580 close4:goto2460
2590 printtab(2)"[CRSR-DOWN][CTRL 9]ond
     erbreking"
2600 gosub1210:printtab(2)"[CRSR-DOWN]t
     yp[SPACE]'←'[SPACE]voor[SPACE]menu
     ":printtab(2)"typ[SPACE]'d'[SPACE]
     voor[SPACE]doorgaan"
2610 getkeyw$
2620 ifw$="←"then50
2630 ifw$="d"then2490:elsegosub1210:got
     o2610
2640 rem "[10xDEL]
2650 rem "[10xDEL]
2660 rem "[10xDEL][10xSPACE]einde[SPACE
     ]listing
2670 rem "[10xDEL]
2680 rem "[10xDEL][10xSPACE]door[SPACE]
     paul[SPACE]siem
2690 rem "[10xDEL]
2700 rem "[10xDEL][10xSPACE]uit[SPACE]r
     ozendaal
2710 rem "[10xDEL]
2720 rem "[10xDEL][10xSPACE]-----------
     ---

     ** EINDE LISTING adresbak 128
```

REGEL 1	3	REGEL 50	209
REGEL 2	35	REGEL 60	219
REGEL 3	33	REGEL 70	5
REGEL 4	143	REGEL 80	170
REGEL 10	152	REGEL 90	5
REGEL 20	144	REGEL 100	219
REGEL 30	174	REGEL 110	238
REGEL 40	56	REGEL 120	167

```
REGEL 130   187
REGEL 140   113
REGEL 150   201
REGEL 160   39
REGEL 170   214
REGEL 180   187
REGEL 190   48
REGEL 200   21
REGEL 210   154
REGEL 220   10
REGEL 230   152
REGEL 240   83
REGEL 250   24
REGEL 260   21
REGEL 270   103
REGEL 280   42
REGEL 290   162
REGEL 300   33
REGEL 310   212
REGEL 320   238
REGEL 330   110
REGEL 340   238
REGEL 350   83
REGEL 360   67
REGEL 370   21
REGEL 380   132
REGEL 390   36
REGEL 400   238
REGEL 410   41
REGEL 420   193
REGEL 430   66
REGEL 440   227
REGEL 450   240
REGEL 460   227
REGEL 470   72
REGEL 480   107
REGEL 490   229
REGEL 500   255
REGEL 510   245
REGEL 520   21
REGEL 530   83
REGEL 540   74
REGEL 550   20
REGEL 560   90
REGEL 570   21
REGEL 580   45
REGEL 590   124
REGEL 600   50
REGEL 610   83
REGEL 620   150
REGEL 630   88
REGEL 640   21
REGEL 650   142
REGEL 660   103
REGEL 670   2
REGEL 680   83
REGEL 690   141
REGEL 700   200
REGEL 710   56
REGEL 720   10
REGEL 730   198
REGEL 740   21
REGEL 750   160
REGEL 760   146
REGEL 770   145
REGEL 780   104
REGEL 790   127
REGEL 800   78
REGEL 810   34
REGEL 820   178
REGEL 830   182
REGEL 840   182
REGEL 850   186
REGEL 860   83
REGEL 870   237
REGEL 880   88
REGEL 890   21
REGEL 900   142
REGEL 910   103
REGEL 920   68
REGEL 930   181
REGEL 940   232
REGEL 950   242
REGEL 960   206
REGEL 970   83
REGEL 980   141
REGEL 990   88
REGEL 1000   182
REGEL 1010   22
REGEL 1020   83
REGEL 1030   152
REGEL 1040   57
REGEL 1050   217
REGEL 1060   125
REGEL 1070   83
REGEL 1080   22
REGEL 1090   21
REGEL 1100   103
REGEL 1110   153
REGEL 1120   222
REGEL 1130   159
REGEL 1140   169
REGEL 1150   213
REGEL 1160   3
REGEL 1170   218
REGEL 1180   3
REGEL 1190   213
REGEL 1200   182
REGEL 1210   137
REGEL 1220   142
REGEL 1230   46
REGEL 1240   51
REGEL 1250   166
REGEL 1260   40
REGEL 1270   51
REGEL 1280   36
REGEL 1290   57
REGEL 1300   84
REGEL 1310   163
REGEL 1320   142
REGEL 1330   38
REGEL 1340   63
REGEL 1350   234
REGEL 1360   159
REGEL 1370   142
REGEL 1380   131
REGEL 1390   25
REGEL 1400   232
REGEL 1410   198
REGEL 1420   86
REGEL 1430   99
REGEL 1440   86
REGEL 1450   187
REGEL 1460   222
REGEL 1470   88
REGEL 1480   187
REGEL 1490   138
REGEL 1500   71
REGEL 1510   128
REGEL 1520   133
REGEL 1530   220
REGEL 1540   142
REGEL 1550   226
REGEL 1560   181
REGEL 1570   152
REGEL 1580   6
REGEL 1590   191
REGEL 1600   219
REGEL 1610   251
REGEL 1620   81
REGEL 1630   142
REGEL 1640   203
REGEL 1650   9
REGEL 1660   170
REGEL 1670   152
REGEL 1680   119
REGEL 1690   219
REGEL 1700   81
REGEL 1710   41
REGEL 1720   142
REGEL 1730   212
REGEL 1740   100
REGEL 1750   113
REGEL 1760   100
REGEL 1770   201
REGEL 1780   236
REGEL 1790   102
REGEL 1800   142
REGEL 1810   138
REGEL 1820   179
REGEL 1830   218
REGEL 1840   240
REGEL 1850   142
REGEL 1860   152
REGEL 1870   138
REGEL 1880   219
REGEL 1890   142
REGEL 1900   198
REGEL 1910   216
REGEL 1920   246
REGEL 1930   209
REGEL 1940   142
REGEL 1950   51
REGEL 1960   210
REGEL 1970   230
REGEL 1980   118
REGEL 1990   131
REGEL 2000   118
REGEL 2010   219
REGEL 2020   254
REGEL 2030   120
REGEL 2040   212
REGEL 2050   81
REGEL 2060   142
REGEL 2070   24
REGEL 2080   81
REGEL 2090   192
REGEL 2100   148
REGEL 2110   227
REGEL 2120   42
REGEL 2130   12
REGEL 2140   21
REGEL 2150   103
REGEL 2160   154
REGEL 2170   251
REGEL 2180   219
REGEL 2190   195
REGEL 2200   159
REGEL 2210   93
REGEL 2220   151
REGEL 2230   52
REGEL 2240   117
REGEL 2250   234
REGEL 2260   48
REGEL 2270   87
REGEL 2280   173
REGEL 2290   98
REGEL 2300   112
REGEL 2310   195
REGEL 2320   159
REGEL 2330   250
REGEL 2340   35
REGEL 2350   92
REGEL 2360   137
REGEL 2370   239
REGEL 2380   48
REGEL 2390   87
REGEL 2400   175
REGEL 2410   112
REGEL 2420   246
REGEL 2430   204
REGEL 2440   175
REGEL 2450   234
REGEL 2460   219
REGEL 2470   142
REGEL 2480   18
REGEL 2490   51
REGEL 2500   210
REGEL 2510   241
REGEL 2520   129
REGEL 2530   142
REGEL 2540   129
REGEL 2550   230
REGEL 2560   9
REGEL 2570   131
REGEL 2580   99
REGEL 2590   120
REGEL 2600   209
REGEL 2610   21
REGEL 2620   103
REGEL 2630   162
REGEL 2640   121
REGEL 2650   121
REGEL 2660   248
REGEL 2670   121
REGEL 2680   13
REGEL 2690   121
REGEL 2700   11
REGEL 2710   121
REGEL 2720   239

READY.
```

Lijn 128

Een korte naam vor een uitgebreid programma. Probeer om een lijn die over het scherm loopt zolang mogelijk te laten worden, zonder de rand, obstakels, of de lijn zelf te raken. Op het scherm staan een aantal hartjes die moeten worden "opgegeten". Het level wordt beindigt door als laatste het rondje te raken, waarna een hoger niveau zich aandient. Er zijn 6 verschillende niveau's. Hoe langer de lijn wordt des te hoger is de te verdienen ɔonus. Voor de wat verder gevorderde programmeurs: ·let programma maakt gebruik van een machinetaal nterrupt routine om de laatste twee joystick bewegingen e bufferen. (buffer op $1310, routine van af $1312). Zonder deze routine zou de joystick steeds in de jewenste stand moeten worden gehouden precies op ıet gewenste moment dat de trage basic daarom /raagt. Hierdoor zou alle soepelheid uit het spel verloren jaan. Dit is heel makkelijk aan te tonen door in regel I70 de "J=peek(JB)" te vervangen door "J=joy(2)". Het ɔrogramma maakt gebruik van een 40 koloms scherm.

```
10 goto 580:rem"[2xSHIFT *][SPACE]l[S
   PACE]i[SPACE]j[SPACE]n[SPACE][2xSH
   IFT *][SPACE]c128[SPACE][4xSHIFT *
   ]IU[37xSHIFT *]I[2xSHIFT -]←[6xSHI
   FT *][SPACE](c)[SPACE]1986[2xSPACE
   ]yprQsoftware[SPACE][6xSHIFT *]K[S
   HIFT -]J[38xSHIFT *]K
20 do:q1=int(rnd(1)*1000+1024):loop u
   ntil peek(q1)=32:return
30 pen pp$:p=32:s=s+25:nn=nn-1:return
40 pen pp$:p=32:s=s+25:nn=nn-1:if nn<
   4 then return
50 gosub 20:if nn=4 then poke q1,87:p
   oke q1+qc,14:return:else poke q1,8
   3:poke q1+qc,13:return
60 pen pp$:p1=p:p=32:s=s+25:nn=nn-1:i
   f p1=83 then return
70 nk=nk-1:if nk<0 then return
80 gosub 20:poke q1,83:poke q1+qc,13:
   if nk=0 then gosub 20:poke q1,87:p
   oke q1+qc,14:return:else return
90 pen pp$:p1=p:p=32:s=s+25:nn=nn-1:i
   f p1=83 then return
100 nk=nk-1:if nk<0 then return
110 gosub 20:poke q1,83:poke q1+qc,13:
    gosub 20:if nk=0 then poke q1,87:p
    oke q1+qc,14:return:else poke q1,9
    0:poke q1+qc,13:return
120 pen pp$:p1=p:p=32:s=s+25:nn=nn-1:i
    f p1=83 then return
130 nk=nk-1:if nk<>0 then return
140 for i=1 to 5:gosub 20:poke q1,83:p
    oke q1+qc,13:next:gosub 20:poke q1
    ,87:poke q1+qc,14:return
150 do:gosub 310: 1:poke jb+1,0:poke j
    b,0:sr=1:do:get k$:loop until k$="
    "
160 do:t=0:do:t=t+1
170 : : if dm=0 then j=peek(jb):poke j
    b,0:rr=zr(r,j):else rr=r+int(1.3*r
    nd(1)-.15):if rr=0 then rr=4:else
    if rr=5 then rr=1
180 : : q1=q+l(rr+10):if dm=0 then dx=
    -1:else p=peek(q1):dx=(t>tx or p=3
    2 or p=83 or p=87 or p=90)
190 : loop until dx:poke q,z(r,rr):pok
    e q+qc,7:q=q1:r=rr:p=peek(q):s=s+1
    :sr=sr*1.01
200 : print using "[HOME][CTRL 9]score
    :#####[SPACE]bonus:####";s,int(sr/
    2)
210 : if p=83 or p=90 then on nv gosub
     30,30,40,60,90,120
220 : get k$:if k$=chr$(136) then p=87
    :dm=1
230 loop until p<>32:qy=int((q-1024)/4
    0):qx=q-1024-40*qy:if qx<1 or qx>3
    8 or qy<1 or qy>23 then r=r+2:if r
    >4 then r=r-4
240 if dm=0 and p=87 and nn=0 then gos
    ub 260:else if dm<=0 then gosub 28
    0
250 loop until dm or lv=0:return
260 scnclr:char 1,10,6,"[CTRL 6]U[17xS
    HIFT *]I":char 1,10,7,"[SHIFT -][S
    PACE][COM 6]g[SPACE]o[SPACE]e[SPAC
    E]d[3xSPACE]z[SPACE]o[SPACE]![SPAC
    E][CTRL 6][SHIFT -]":char 1,10,8,"
    J[17xSHIFT *]K":nv=nv+1:if nv>nx t
    hen s=s+100:nv=nx
270 s=s+int(sr/2):char 1,5,11:print "[
    COM 8]score:";s;"[2xSPACE]volgend[
    SPACE]nivo:";nv:pen"t7o4qc.figqao5
    c.fidqcifeqfo4$bag.f": 2:return
280 poke q,42:poke q+qc,10:sound 1,800
    0,100,1,4000,40,2,2048: 3:lv=lv-1:
    if lv>0 then return
290 scnclr:char 1,8,7,"[CTRL 5]U[23xSH
    IFT *]I":char 1,8,8,"[SHIFT -][SPA
    CE][COM 8]het[SPACE]spel[SPACE]is[
    SPACE]afgelopen[SPACE][CTRL 5][SHI
    FT -]":char 1,8,9,"J[23xSHIFT *]K"
300 char 1,15,12:print"[COM 8]score:";
    s:pen"t1u15o3qfao4co3afahcc.fqro4u
    4": 2:return
310 scnclr:color 5,11:poke 248,192:pri
    nt using "[CTRL 9][2xSPACE]"+tx$(-
    dm)+"[4xSPACE]nivo:#[SPACE]lijnen:
    #[SPACE][CTRL 0]";nv,lv;
320 for i=1 to 23:print"[CRSR-LEFT][CT
    RL 9][2xSPACE][CTRL 0]":next:print
    "[CRSR-LEFT][CTRL 9][41xSPACE][CTR
    L 0]"
330 poke 248,0:poke q,81:poke q+qc,7
340 if nv<3 then i1=10:else if nv=5 th
    en i1=1:else i1=5
350 if nv<4 then p=83:nk=0:else p=90:n
    k=5
360 for i=1 to i1:gosub 20:poke q1,p:p
    oke q1+qc,13:next:nn=10
370 if nv<3 then gosub 20:poke q1,87:p
    oke q1+qc,14
380 if nv>1 then for i=1 to 5+5*nv:gos
    ub 20:poke q1,160:poke q1+qc,10:ne
    xt
390 return
400 print chr$(147)chr$(142)chr$(11):n
    x=6:tx=10:qc=54272:pp$="t0a":point
```

```
    er 4,6,0,9,12,0:key 7,chr$(136)
410 for i=1 to 4:for j=1 to 4:read z(i
    ,j):next:next:data 64,73,81,75,74,
    93,75,81,81,85,64,74,85,81,73,93
420 for r=1 to 4:for j=0 to 8:read zr(
    r,j):next:next
430 data 1,4,4,1,2,2,2,3,4,2,4,1,1,1,2
    ,3,3,3
440 data 3,4,4,1,2,2,2,3,4,4,4,1,1,1,2
    ,3,3,3
450 pot 15:for q=4864 to 4945:read p:p
    oke q,p:next:sys 4933:jb=4880:rem
    ml joy buffer rout
460 data 0,0,0,0,0,4,2,3,0,6,8,7,0,5,1
    ,0
470 data 0,0,162,1,173,16,19,240,7,173
    ,17,19,208,36,240,15
480 data 160,0,173,17,19,140,17,19,141
    ,16,19,208,2,162,0,173
490 data 0,220,41,15,168,185,0,19,240,
    8,205,16,19,240,3,157
500 data 16,19,76,101,250,120,169,18,1
    41,20,3,169,19,141,21,3,88,96
510 read ll$,l$:ll$=ll$+l$:ll=len(ll$)
520 data 1aaaf4bbbbc1d2bbbe1aad2e1af4b
    bbbc1d2bbbbe1f4c1aj4bbbc1d2bbbbe1a
    f4bbbbc1d2i1h2i1h2i1j4bbc1d2bbbbe1
    aaaaaaaaaaaaaaaa
530 data h2bj3aaaaaaaaaaaaaaaaaaaaaaaa
    aaaaaaaaaaaaaai4g1aaaaaaaaaaaaaaaa
    aaaaaaaaaaaaaaaaaaaaa3aaaa
540 dim l(14):for i=1 to 14:read l(i):
    next:data 64,93,112,110,109,125,85
    ,73,74,75,1,40,-1,-40
550 read mm$:mm=len(mm$)
560 data o4qc.figqao5c.fidqco4$bafgeif
    gafq.cirqc.figqao5c.fidqcifeqfo4$b
    ag.firo5ccfcccqcicccо4fffqfifao5co
    4afaqcc.f
570 tx$(0)=">[SPACE]houdt[SPACE]u[SPAC
    E]gereed[SPACE]<":tx$(1)="[SPACE]d
    emonstratiespel[SPACE]":return
580 color 0,1:color 4,11:color 5,5:scn
    clr:gosub 400
590 do:color 5,8:scnclr:color 5,7:char
     1,28,2,"(c)[SPACE]1986":char 1,26
    ,3,"yprQsoftware":char 1,0,8
600 print"[2xCRSR-DOWN][COM 8][2xSPACE
    ]laat[SPACE]de[SPACE]lijn[SPACE]zo
    [SPACE]lang[SPACE]mogelijk[SPACE]w
    orden":print"[10xSPACE]eet[SPACE]d
    aarbij[SPACE]alle[SPACE][COM 6]S[C
    OM 8][SPACE]op":print"[14xSPACE]ui
    tgang[SPACE]is[SPACE][COM 7]W"
610 print using "[2xCRSR-DOWN][CTRL 4]
    [15xSPACE]score[SPACE]:[SPACE]####
    #";s:print using "[CRSR-DOWN][COM
    3][7xSPACE]hoogste[SPACE]score[SPA
    CE]:[SPACE]#####";hs:print "[CRSR-
    DOWN][COM 8][16xSPACE]nivo[SPACE]:
    [2xSPACE]1"
620 print "[CRSR-DOWN][COM 6][19xSPACE
    ][10xCOM @]":print "[COM 6][2xSPAC
    E]spel[SPACE]starten[SPACE]dmv[SPA
    CE][CTRL 9][SPACE]vuurknop[SPACE][
    CTRL 0][SPACE](port[SPACE]2)":prin
    t "[COM 5][2xSPACE]spel[SPACE]afbr
    eken[SPACE]met[SPACE][CTRL 9][SPAC
    E]f7[SPACE][CTRL 0]"
```

```
630 nv=1:color 5,16:q=1263:q1=1:j1=0:i
    =9:l=1:m=1:k=4:rsppos 10:pen"t4":v
    ol 15
640 do:j=joy(2):if jand128 then exit
650 : if j<>j1 then tron:j1=j:if j=1 t
    hen nv=nv+1:if nv>nx then nv=1
660 : : if j=5 then nv=nv-1:if nv<1 th
    en nv=nx
670 : : char 1,23,19,str$(nv):troff
680 : i=i+1:if i>4 then tron:i=1:p=ins
    tr("abcdefghij1234",mid$(ll$,l,1))
    :l=l+1:if p>10 then q1=l(p):i=9:el
    se q=q+q1:poke q,l(p):troff
690 : k=k+1:if k>8 then tron:k=0:m$=mi
    d$(mm$,m,1):m=m+1
700 : :  p=instr("oiq",m$):if p then t
    ron:pen m$:m$=mid$(mm$,m,1):m=m+1:
    if p=1 then penm$:m$=mid$(mm$,m,1)
    :m=m+1:else k1=12/p:troff
710 : :  p=instr(".$#",m$):if p then m
    $=m$+mid$(mm$,m,1):m=m+1
720 : pen m$:k=k1:if p=1 then k=k*2/3:
    troff
730 : if m>mm+25 then m=1
740 loop until l>ll:q=1524:r=1:s=0:lv=
    3:pen"o4q":rsppos 15:vol 4
750 if j<128 then dm=-1:gosub150:else
    dm=0:gosub150:if s>hs then hs=s
760 loop
```

REGEL 10	188	REGEL 390	142
REGEL 20	16	REGEL 400	64
REGEL 30	210	REGEL 410	229
REGEL 40	135	REGEL 420	79
REGEL 50	35	REGEL 430	250
REGEL 60	95	REGEL 440	254
REGEL 70	54	REGEL 450	46
REGEL 80	69	REGEL 460	59
REGEL 90	95	REGEL 470	230
REGEL 100	54	REGEL 480	73
REGEL 110	78	REGEL 490	25
REGEL 120	95	REGEL 500	178
REGEL 130	231	REGEL 510	177
REGEL 140	119	REGEL 520	93
REGEL 150	191	REGEL 530	52
REGEL 160	239	REGEL 540	52
REGEL 170	178	REGEL 550	157
REGEL 180	152	REGEL 560	139
REGEL 190	236	REGEL 570	83
REGEL 200	45	REGEL 580	139
REGEL 210	30	REGEL 590	139
REGEL 220	168	REGEL 600	55
REGEL 230	165	REGEL 610	3
REGEL 240	165	REGEL 620	249
REGEL 250	117	REGEL 630	124
REGEL 260	83	REGEL 640	96
REGEL 270	64	REGEL 650	184
REGEL 280	145	REGEL 660	157
REGEL 290	243	REGEL 670	226
REGEL 300	128	REGEL 680	68
REGEL 310	76	REGEL 690	231
REGEL 320	34	REGEL 700	190
REGEL 330	11	REGEL 710	251
REGEL 340	226	REGEL 720	86
REGEL 350	19	REGEL 730	69
REGEL 360	137	REGEL 740	101
REGEL 370	187	REGEL 750	208
REGEL 380	134	REGEL 760	236

Lichtkrant c 16

Uit Buntschoten ons volgende programma, de heer A. Bongiovanni heeft een lichtkrant voor de c 16 gemaakt. Hij presteerde het om in een paar regels een goed werkende lichtkrant te maken.

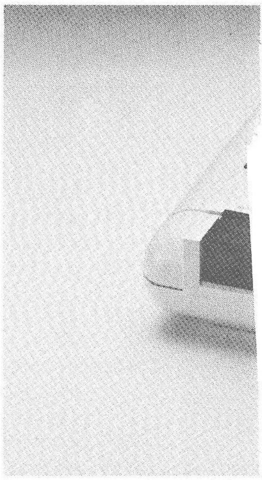
```
  1 print"[SHIFT CLR][2xHOME]"
  2 rem********************************
    ****
  3 rem*       lichtkrant voor c16/+4
    *
  4 rem*            door f.ten boske
    *
  5 rem*                 zeist (c)86
    *
  6 rem********************************
    ****
 50 color0,1:color1,2:color4,1:graphic
    4,1
 60 draw1,54,50to86,50to70,70
 70 draw1,70,70to54,50
 80 draw1,64,50to70,28to76,50
 90 draw1,62,60to58,80to70,69
100 draw1,78,60to82,80to70,69
110 char1,10,7,"star":char1,21,7,"soft
120 char1,17,7,"*":char1,7,14,"****by[
    SPACE]f.ten[SPACE]boske****"
130 getkeya$:graphic0
140 forl=1to1100:next
150 dima$(50)
160 forx=1tot:a$(x)="[CTRL 5]":next:t=
    0
170 print"[SHIFT CLR][CRSR-DOWN][CTRL
    2]"
180 printspc(11)"W[3xSPACE]W[2xSPACE]W
    W[SPACE]W[SHIFT SPACE]W[SPACE]WWW
190 printspc(11)"W[3xSPACE]W[SPACE]W[3
    xSPACE]W[SPACE]W[2xSPACE]W
200 printspc(11)"W[3xSPACE]W[SPACE]W[3
    xSPACE]WWW[2xSPACE]W
210 printspc(11)"Q[3xSPACE]Q[SPACE]Q[3
    xSPACE]Q[SPACE]Q[2xSPACE]Q[3xSHIFT
     SPACE]
220 printspc(11)"QQQ[SPACE]Q[2xSPACE]Q
    Q[SPACE]Q[SPACE]Q[2xSPACE]Q
225 forl=1to599:next:print"[3xCRSR-DOW
    N]"
230 printspc(10)"W[SHIFT SPACE]W[SHIFT
     SPACE]WW[SPACE][2xSHIFT SPACE]W[2
    xSHIFT SPACE]WW[2xSHIFT SPACE]WWW
240 printspc(10)"W[SHIFT SPACE]W[SHIFT
     SPACE]W[SHIFT SPACE]W[SHIFT SPACE
    ]W[SHIFT SPACE]W[SHIFT SPACE]W[SHI
    FT SPACE]W[SHIFT SPACE][SPACE]W
250 printspc(10)"WW[2xSHIFT SPACE]WW[2
    xSHIFT SPACE]WWW[SHIFT SPACE]W[SHI
    FT SPACE]W[2xSHIFT SPACE]W
260 printspc(10)"Q[SPACE]Q[SPACE]Q[SPA
    CE]Q[SPACE]Q[SPACE]Q[SPACE]Q[SPACE
    ]Q[2xSPACE]Q
270 printspc(10)"Q[SPACE]Q[SPACE]Q[SHI
    FT SPACE]Q[SPACE]Q[SPACE]Q[SPACE]Q
    [SPACE]Q[2xSPACE]Q
280 forl=1to999:next:print"[SHIFT CLR]
    "
290 print:print:forz=1to1000:next
300 print"[2xSPACE]maximaal[SPACE]50[S
    PACE]regels."
310 print"[2xSPACE]niet[SPACE]meer[SPA
    CE]dan[SPACE]30[SPACE]tekens[SPACE
    ]per[SPACE]regel
320 print"[2xSPACE]geen[SPACE]komma[SP
    ACE]of[SPACE]dubbelpunt.
330 print"[2xSPACE]na[SPACE]elke[SPACE
    ]regel[SPACE]<return>.
340 print"[2xSPACE]laatste[SPACE]regel
    [SPACE]<*>.
350 print"[CRSR-DOWN][6xSPACE]1[3xSPAC
    E]5[4xSPACE]10[4xSPACE]15[3xSPACE]
    20[3xSPACE]25[3xSPACE]30
360 print"
370 t=t+1:printt;tab(4);
380 inputa$(t)
390 ifa$(t)="*"thena$(t)="[CTRL 2][SPA
    CE]":t=t-1:goto460
400 ift=50then 450
410 iflen(a$(t))<1thent=t-1:goto370
420 iflen(a$(t))>32thenprint"te[SPACE]
    lang,opnieuw[SPACE]invoeren.":t=t-
    1:goto370
430 ift/6=int(t/6)thenprinttab(19)"[CR
    SR-DOWN]laatste[SPACE]regel[SPACE]
    <s>":goto360
440 goto 370
460 print"[SHIFT CLR][4xCRSR-DOWN]"
470 print"[SPACE]UCCCCCCCCCCCCCCCCCCC
    CCCCCCCCCCCCCCCI[SPACE][SHIFT SPAC
    E]
480 printtab(1)"B[35xSPACE]B
490 printtab(1)"B[23xSPACE][3xSHIFT SP
    ACE][9xSPACE]B
500 print"[SPACE]JCCCCCCCCCCCCCCCCCCCC
    CCCCCCCCCCCCCCCK
520 print"[10xCRSR-DOWN][2xSPACE]onder
    breken:[SPACE]toets[SPACE]'s'[SPAC
    E]in
540 for y=1tot
550 forx=1tolen(a$(y))
560 print"[HOME][6xCRSR-DOWN]
570 for z=1to80:next
580 printtab(36-x)"[CTRL 9]";left$(a$(
    y),x)
590 nextx
600 forx=(35-len(a$(y)))to4 step-1
610 print"[HOME][6xCRSR-DOWN]
620 forz=1to80:next
630 printtab(x)a$(y);"-"
640 getz$:ifz$="s"then x=4:y=t:s=1:pri
    nt"[SPACE]"
650 nextx:ifs=1then 720
660 forx=1tolen(a$(y))
670 print"[HOME][6xCRSR-DOWN]
680 forz=1to80:next
690 printtab(4)right$(a$(y),(len(a$(y)
    )-x));"-
700 nextx
710 getz$:ifz$="s"theny=t:s=1:print"[S
    HIFT CLR]"
720 nexty:ifs=1then 750
730 goto540
750 s=0:print"[4xCRSR-DOWN][CTRL 4]
```

```
 760 print"[2xSPACE]geheel[SPACE]nieuwe
     [SPACE]tekst....1
 770 print"[2xSPACE]zelfde[SPACE]tekst[
     SPACE]opnieuw...2
 780 print"[2xSPACE]tekst[SPACE]verande
     ren.......3
 790 print"[2xSPACE]stoppen...........
     ....4
 800 getz$:onval(z$)+1goto800,160,460,8
     20
 810 print"[CRSR-DOWN][2xSPACE]einde[SP
     ACE]lichtkrant.":forl=1to1200:next
     :color4,2:sys62116
 820 gosub840:goto460
 840 e=1:d=10
 850 print"[SHIFT CLR]";"[3xSPACE]nr[SP
     ACE]tekst[CRSR-DOWN]
 860 goto900
 870 e=e+10:d=e+9
 880 ife>tthenreturn
 890 goto850
 900 ifd>tthend=t
 910 forx=etod
 920 printtab(1)x;"[2xSPACE]";a$(x):nex
     tx
 930 print"[2xCRSR-DOWN][2xSPACE]regel[
     SPACE]veranderen....1
 940 print"[2xSPACE]regel[SPACE]toevoeg
     en.....2
 950 print"[2xSPACE]regel[SPACE]verwijd
     eren...3
 960 print"[2xSPACE]okee[SPACE]zo......
     .......4
 970 getk$:onval(k$)+1 goto980,1100,110
     0,1320,990
 980 goto970
 990 goto870
1010 print:input"[2xSPACE]regelnr.";r
1020 ifr<eorr>dthenprint"[SPACE]";:fors
     p=1to39:print"[SPACE]";:nextsp:pri
     nt"[SPACE]";:goto1010
1030 print"
1040 print"is:[SPACE]"tab(7);a$(r)
1050 input"wordt";a$(r)
1060 iflen(a$(r))<1then1050
1070 iflen(a$(r))>30 then1050
1080 goto850
1100 ift<50then1180
1110 print"[CRSR-DOWN]maximum[SPACE]aan
     tal[SPACE]regels[SPACE]bereikt.[SP
     ACE]bij[SPACE]"
1120 print"toevoeg[SPACE]en/of[SPACE]tu
     ssenvoegen[SPACE]vervalt
1170 t=t-1
1180 input"[CRSR-DOWN]toevoegen[SPACE]n
     a[SPACE]welk[SPACE]regelnummer:[SP
     ACE]";r
1190 ifr<e-1or r>d thenprint"[CRSR-UP]"
     ;:forsp=1to39:print"[CRSR-UP]";:ne
     xt sp:print"[2xCRSR-UP]":goto1180
1200 ifr=50then r=49
1210 forx=ttor+1step-1
1220 a$(x+1)=a$(x):nextx
1230 a$(r+1)= "[SPACE]"
1240 print"[SPACE]nieuwe[SPACE]regel
1250 print"
1260 input"tekst:[SPACE]";a$(r+1)
1270 iflen(a$(r+1))<1then1250
1280 iflen(a$(r+1))>30 then1250
1290 d=d+1:t=t+1
1300 goto850
1320 input"[CRSR-DOWN]welke[SPACE]regel
     [SPACE]verwijderen:";r
1330 ifr<eorp>dthen print"[CTRL 8]";:fo
     rsp=1to39:print"[CTRL 2][CTRL 1]";
     :nextsp:print"[2xSPACE]":goto1320
1340 print:printtab(1)r;"[SPACE]";a$(r)
1350 print"[CRSR-DOWN][3xSPACE]zeker[SP
     ACE]weten?[SPACE](j/n)
1355 inputz$
1360 ifz$="j"then 1380
1370 ifz$="n"then 850
1380 forx=rtot-1
1390 a$(x)=a$(x+1):nextx
1400 a$(t)="[SPACE]"
1410 d=d-1:t=t-1:goto850
```

** EINDE LISTING lichtkrant c16

REGEL 1	150	REGEL 470	162
REGEL 2	77	REGEL 480	60
REGEL 3	85	REGEL 490	28
REGEL 4	230	REGEL 500	249
REGEL 5	116	REGEL 520	149
REGEL 6	77	REGEL 540	181
REGEL 50	127	REGEL 550	131
REGEL 60	125	REGEL 560	52
REGEL 70	218	REGEL 570	134
REGEL 80	128	REGEL 580	14
REGEL 90	135	REGEL 590	218
REGEL 100	139	REGEL 600	111
REGEL 110	114	REGEL 610	52
REGEL 120	11	REGEL 620	134
REGEL 130	71	REGEL 630	120
REGEL 140	210	REGEL 640	176
REGEL 150	161	REGEL 650	21
REGEL 160	186	REGEL 660	131
REGEL 170	134	REGEL 670	52
REGEL 180	27	REGEL 680	134
REGEL 190	246	REGEL 690	239
REGEL 200	205	REGEL 700	218
REGEL 210	178	REGEL 710	203
REGEL 220	69	REGEL 720	25
REGEL 225	1	REGEL 730	34
REGEL 230	81	REGEL 750	13
REGEL 240	122	REGEL 760	166
REGEL 250	81	REGEL 770	227
REGEL 260	68	REGEL 780	165
REGEL 270	228	REGEL 790	248
REGEL 280	101	REGEL 800	201
REGEL 290	133	REGEL 810	13
REGEL 300	124	REGEL 820	134
REGEL 310	106	REGEL 840	185
REGEL 320	7	REGEL 850	9
REGEL 330	98	REGEL 860	34
REGEL 340	10	REGEL 870	159
REGEL 350	37	REGEL 880	10
REGEL 360	187	REGEL 890	38
REGEL 370	210	REGEL 900	197
REGEL 380	143	REGEL 910	184
REGEL 390	46	REGEL 920	202
REGEL 400	54	REGEL 930	31
REGEL 410	199	REGEL 940	238
REGEL 420	178	REGEL 950	44
REGEL 430	34	REGEL 960	18
REGEL 440	35	REGEL 970	1
REGEL 460	180	REGEL 980	41

De datarecorder wordt veel gebruikt voor het opslaan van programma's en data. Een leuke, andere toepassing ervan is er muziek mee opnemen. Geschreven door Marc de Hingh.



"Musicbox"

Muziek met de Datasette - Commodore 64

Met wat eenvoudige trucs kan men tonen met de datarecorder op tape zetten. Door het bandje vervolgens in een gewone cassetterecorder te leggen, kan men dan de opgenomen muziek beluisteren.

De Commodore-datarecorder werkt als het ware digitaal. Daarmee bedoel ik dat hij de computer alleen maar kan doorgeven of er op de plaats op de band waar de kop langs gaat, wel of geen signaal staat - met andere woorden: hoge dan wel lage spanning - oftewel 1 of 0.
De andere kant op werkt hetzelfde: de computer kan de schrijf-pin van de cassettepoort 'hoog' of 'laag' maken. Dit zal resulteren in een hoge of lage spanning op de schrijfkop van de recorder - wel of geen signaal op de band. Alle waarden tussen hoge en lage spanning zijn dus uitgesloten. Dientengevolge kan men, wanneer men muziek naar de datarecorder wil sturen, slechts op één volume werken. Dit beperkt ons enigszins in de mogelijkheden, maar er kunnen toch aardige resultaten bereikt worden.

Cassettepoort

Op pagina 397 van de Reference Guide vinden we de verdeling van de verschillende pinnen van de cassettepoort:

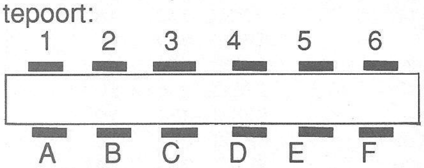


fig. 1. de cassettepoort.

Pin:	Type:
A-1	GND
B-2	+5V
C-3	Cassette Motor
D-4	Cassette Read
E-5	Cassette Write
F-6	Cassette Sense

Naast een aarde-verbinding en de 5 Volt-voeding van de recorder zien we de lees- en schrijfpinnen en nog een tweetal pinnen voor de controle van de motor van de datasette.
De uitgang die voor ons nu het interessantst is, de cassette write-pin, is direct verbonden met de 6510 microprocessor.
We kunnen heel gemakkelijk de write-pin 'hoog' of 'laag' maken door een 1 resp. 0 op pin P3 van de I/O-poort van de microprocessor te zetten. Dit kan door 1 of 0 te poken in bit 3 van het I/0-register op adres 1 in het geheugen.

Een toon

Om een toon te genereren moeten we voortdurend wisselen tussen een 1 en een 0 op de genoemde lijn.

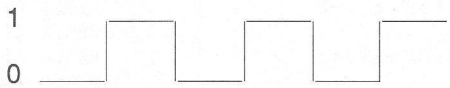


fig. 2. schema van een blokgolf.

We krijgen dan immers een blokgolfvorm als in fig. 2
Dat kunnen we gemakkelijk testen in basic. Typ het hieronderstaande programmaatje over, leg een lege cassette in de datarecorder en druk op record en play. Run het programma vervolgens. Na een tijdje breaken en de recorder stoppen

Als we het opgenomen geluid naderhand met een normale recorder beluisteren, horen we een laag gebrom :het bewijs dat er inderdaad iets op-

```
10 POKE 1, PEEK(1) OR 8: REM
           bit 3 = 1
20 POKE 1, PEEK(1) AND 247: REM
           bit 3 = 0
30 GOTO 10
```

(progr .1.)

genomen is. Willen we echter hogere tonen, dan blijkt dat de basic tekortschiet. Voor hogere tonen moeten we namelijk vaker per seconde (hogere frequentie!) heen en weer springen tussen 0 en 1 en daarvoor hebben we een hogere snelheid nodig: basic is te langzaam!
De oplossing is machinetaal. Als u (nog) niet ingewijd bent in de geheimen van de machinetaal, laat u zich dan toch vooral niet afschrikken: de gebruikte routine is extreem kort en behoeft natuurlijk niet perse volledig begrepen te worden om toegepast te worden in basic-programma's.

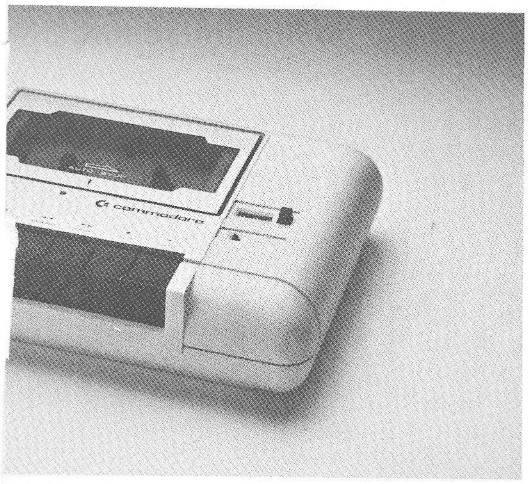
Machinetaalroutine

In assembly ziet de benodigde routine er zo uit:

```
0340 LDA  $01
0342 EOR  #$08
0344  STA  $01
0346 JMP  $EA7E
```

(progr. 2)

Het enige wat deze routine doet, is bit 3 van adres 1 inverteren, d.w.z. van een 0 een 1 maken, van een 1 een 0 maken. Door deze routine een aantal maal per seconde te laten aanroepen, ontstaat een toon en wel één met een frequentie (in Hertz) van de helft van dat aantal maal per secon-



de. (Immers, de routine moet tweemaal doorlopen worden voor een golfcyclus van het geluid.) We zouden nu vanuit een basic-programma met het SYS-commando de machinetaalroutine een aantal maal per seconde kunnen aanroepen. Dit doen wij echter niet, omdat dit nauwelijks sneller zou gaan dan wanneer we gewoon poken, zoals in progr. 1. Daarom doen we het slimmer en maken we gebruik van de zgn. interrupt van de computer.

Interrupt

Zoals u wellicht weet, wordt zestig maal per seconde een routine in het operating-system (kernal) doorlopen om het toetsenbord te scannen en dat soort zaken. Deze routine wordt automatisch iedere éénzestigste seconde aangeroepen d.m.v. een interrupt.
Door de zgn. interrupt-vector op geheugenlocaties 788 en 789 naar onze machinetaalroutine te laten wijzen, wordt de normale interuptroutine afgesneden en wordt in plaats ervan iedere éénzestigste seconde onze eigen routine aangeroepen. De computer gaat dan - u raadt het al - een toon genereren van 60/2=30 Hertz. Maar met alleen tonen van 30 Hertz kunnen we geen melodieën maken, dat is duidelijk. en gelukkig is het dan ook mogelijk om de snelheid van de interrupt en daarmee de hoogte van de toon die gegenereerd wordt te varieren door geschikte waarden te poken in de registers 56324 en 56325. Een lage waarde geeft daarbij een hoge interruptsnelheid, een hoge toon, terwijl hoge waardes lage interruptsnelheden en dus lage tonen veroorzaken.
De precieze te poken waarde is voor elke willekeurige frequentie te berekenen met een constante omrekenigsfactor. Deze blijkt 492630 te zijn, de helft van de klokfrequentie van de C64 (ca. 1 MHz).
Op deze manier kunnen we een melodietje spelen.

Wilhelmus

In een programma, dat de eerste maten van het Wilhelmus naar de datarecorder stuurt, zullen we nu de behandelde technieken toepassen. Vanaf regel 1000 staat de routine om een toon met frequentie F en duur D te genereren. Met behulp van de REM's (die u natuurlijk niet hoeft over te typen) moet het mogelijk zijn het programma te begrijpen en te experimenteren met deze aardige toepassing van de datarecorder.

Marc de Hingh

WILHELMUS

```
10    REM Wilhelmus
100   GOSUB 2000 :REM zet machinetaal in het geheugen
110   PRINT"press rec&play":PRINT"en druk op 'n toets"
120   GET K$:IF K$="" THEN 120 :REM wacht op toets
130   FOR I=I TO 16  :REM 16 tonen
140   READ F,D :REM lees frequentie en duur van een
      toon uit data vanaf regel 3000
150   GOSUB 1000 :REM genereer die toon
160   NEXT 1
170   PRINT"klaar! stop de recorder"
180   END
1000  REM routine voor toon naar data-recorder
1010  REM  f = frequentie in hertz
1020  REM  d = duur van de toon
1030  POKE 56333,1 :REM schakel normale
      interrupt uit
1040  T=492630/F :REM bereken de  interruptsnelheid
1050  H=INT(T/256)  :REM high-byte interruptsnelheid
1060  L=T-256*H :REM low-byte interruptsnelheid
1070  POKE 56324,L:POKE 789,3 :REM poke  bereken
      de snelheid
1080  POKE 788,64:P OKE 789,3 :REM verander
      interruptvecto r
1090  POKE 53265,1 1 :REM maak scherm blank voor
      betere opname kwaliteit
1100  POKE 56333,1 29 :REM start  ..interruptroutine,
      de toon begint  nu
1110  FOR P=1 TO  D*100:NEXT :REM pauze
      voor duur to on
1120  POKE 56333 ,1 :REM stop interruptroutine,
       de toon stopt nu
1130  POKE 53265,27 :REM scherm weer normaal
1140  POKE 788,49:POKE 789,234 :REM  interrupt
      snelheid normaal
1160  POKE 56333,129 :REM start normale interrupt
1170  RETURN
2000  REM lees machinetaal   in het  geheugen
2010  FOR I=832 TO 840:READ X:POKE I,X:NEXT
      :RETURN
2020  DATA 165,173,8,133,1,76,126,234
      :REM data machinetaal-routine
3000  REM data  tonen (freq., duur)
3010  DATA 196,4,262,4,262,4,294,2, 330,2,349,2,294,
      2,330,4
3020  DATA 294,2,330,2,349,4,330,4, 294,2,263, 2,294,
      4,263,8
```

programma 3.

Het ingeven van parameters in een zelfgeschreven machinetaalprogramma voor de Commodore 64 wordt hieronder uitgebreid behandeld door Rene Janssen.

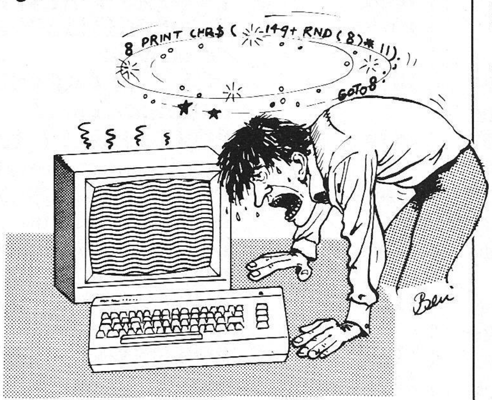
PARAMETER-TRANSPORT

De Basic-opdracht SYS is niet iets voor de beginner, aangezien er in een programma of via een direct commando mee naar het machinetaalniveau gesprongen wordt.

De opdracht Sys wordt meestal gebruikt om het machinetaalprogramma uit te voeren dat begint bij het gegeven adres. In sommige programma's echter komen opdrachten voor als: SYS 647,10,199.
Een ander voorbeeld is de variant op de PRINT-AT instruktie: SYS 49152, kolom, regel, string. Het rijtje gegevens (ook wel **parameters** genoemd) wordt door het machinetaalprogramma zelf verwerkt.
Hoe dat allemaal gebeurt en hoe je parameters aan een zelfgeschreven machinetaalprogramma kunt doorgeven, kun je lezen in het volgende artikel.

De USR-funktie

Zolang je maar 1 parameter nodig hebt, hoef je die speciale SYS-opdracht helemaal niet te gebruiken. Je kan dan de USR-opdracht toepassen. D.m.v. 'USR (waarde)' wordt de gegeven waarde naar de accu voor reële getallen, FAC, gebracht. Daarna wordt het machinetaalprogramma uitgevoerd waarvan het adres in de adressen $0311 (785) en $0312 (786) staat.
B.v.: Het startadres = 49152;
POKE 785,0:POKE 786,192.
D.m.v. 'PRINT USR (kleurcode)' wordt het programma vanaf 49152 uitgevoerd. De kleurcode wordt in FAC geladen. Daarna wordt FAC in een positief 16-bits getal veranderd ($B7F7). De high-byte van dat getal gaat naar $15 (21) en de low-byte naar $14 (20). Vervolgens wordt de low-byte in $14 naar $D020 gebracht en het achtergrondscherm veranderd in de gegeven kleur.
Voordat deze routine echter uitgevoerd kan worden moet de low-



byte van het adres (49152) in $0311 en de high-byte in $0312 geladen worden, dus:
POKE*785,0:POKE*786,192:PRINT USR (kleurcode)

Als i.p.v. 'PRINT USR (waarde)', 'variabele=USR (waarde)' wordt gebruikt, wordt de waarde die na het uitvoeren van de routine in FAC zit, toegekend aan de gegeven variabele. B.v. 'B=USR(23)'.

LISTING 1.

```
C000 20 F7 B7    JSR $B7F7;getal in FAC naar $14 en $15.
C003 A5 14       LDA $14
C005 8D 20 D0    STA $D020;low-byte naar 53280.
C008 60          RTS
```

Het verwerken van parameters

Voor meer dan 1 parameter moet de speciale SYS-opdracht gebruikt worden. Om de verschillende parameters te verwerken wordt gebruik gemaakt van bepaalde ROM-routine's. De belangrijkste hiervan zijn de volgende:

- **JSR $AEFD**: Deze routine controleert of er een komma in een opdracht voorkomt. In de opdracht 'SYS adres, variabele' kan op die manier gekontroleerd worden of er een komma tussen 'adres' en 'variabele' aanwezig is. Is dat niet het geval dan krijg je een 'SYNTAX ERROR'.
- **JSR $AEFA**: Deze routine controleert of er een (-teken staat.
- **JSR $AEF7**: Deze routine controleert of er een)-teken staat.
- **JSR $AEF1**: Deze routine haalt een parameter die tussen haakjes staat.
  B.v.: SYS 49152,(23)
  SYS 2050,(34,456)
  SYS 827,(INT(RND(0)*15))
- **JSR $AD9E**: Deze routine haalt een willekeurige parameter. Deze parameter kan een string zijn, maar kan ook numeriek zijn. Numerieke parameters gaan naar FRC. Is het een string, dan wijst $64/$65 (100/101) naar het adres

vanaf waar de lengte en het beginadres van de string opgeslagen zijn. B.v.:
PEEK(101)*256+PEEK(100)=28
PEEK(28)=lengte string
PEEK(30)*256+PEEK(29)=begin adres string

Voorbeelden van SYS-opdrachten waar gebruik gemaakt kan worden van deze routine zijn:
SYS 49152,PEEK(832)AND32
SYS 49152,"DIT IS EEN VOORBEELD"

° **JSR $AD8A**: Deze routine haalt een parameter en controleert meteen of deze numeriek is.

° **JSR $AD8D**: Deze routine controleert of de binnengehaalde parameter numeriek is. De parameter moet dan eerst binnengehaald zijn met 'JSR $AD9E'.

° **JSR $AD8F**: Deze routine controleert of de binnengehaalde parameter een string is.

° **JSR $B79E**: Deze routine haalt een byte-waarde (0-255) en stopt hem in het X-register.

° **JSR $B7EB**: Deze routine wordt o.a. gebruikt voor de instruktie 'POKE adres, waarde'. Het haalt een 16- en 8-bits waarde. De 16-bits waarde gaat naar $14/$15 (20,21) en de 8-bits waarde gaat naar het X-register.

° **JSR $E1D4**: Deze routine wordt gebruikt voor de 'LOAD-en 'SAVE'-opdracht. Het haalt de filenaam en het devicenummer.

° **JSR $0073**: De CHRGET-routine. Deze routine haalt een karakter uit een Basic-programma.

° **JSR $0079**: De CHRGOT-routine. Deze routine haalt het laatste karakter, uit een Basic-programma dat met de CHRGET-routine is binnengehaald.

Verwerken van hele getallen

Stel dat je een machinetaalprogramma wilt schrijven dat de achter- en voorgrondkleur verandert in de door jouw gegeven waarden. Je zou dan b.v. deze opdracht gebruiken: 'SYS 49152,achter,voor'. 'Achter' en 'voor' zouden dan respectievelijk in de adressen 53280 en 53281 geladen moeten worden. Je kunt dan niet de waarde in FAC gebruiken. Tenminste, niet zonder daar iets aan te veranderen. *Er zijn 2 manieren om gehele getallen te krijgen.*
De eerste manier is:

°1 Haal de waarde binnen met 'JSR $AD9E'. De waarde zit nu in FAC.

°2 Gebruik 'JSR $B7F7'. Deze routine veranderd FAC in een positief 16-bits getal en stopt die waarde in $14/$15.

De tweede manier is: Gebruik of 'JSR $B79E' of 'JSR $B7EB'.
Het volgende voorbeeld heeft deze instruktie: SYS 49152,achtergrond, voorgrond.
Hier wordt 'JSR $B79E' gebruikt om de waarde te halen, waarna die in het X-register gebracht wordt.

```
LISTING 2
C000 20 FD  AE JSR $AEFD ;staat er een komma?
C003 20 9E  B7 JSR $B79E ;haalt waarde (0-255) en stopt hem in X-reg.
C006 8E 20  D0 STX $D020 ;naar 53280.
C009 20 FD  AE JSR $AEFD ;volgende komma.
C00C 20 9E  B7 JSR $B79E ;volgende waarde.
C00F 8E 21  D0 STX $D021 ;naar 53281.
```

Verwerken van reële getallen.

Om nu getallen als 12.3456 of 34.234E+12 te gebruiken kun je de waarde in FAC wel rechtstreeks gebruiken. Je kunt de betreffende waarde nu binnenhalen met: 'JSR $AEF1', 'JSR $AD9E' en 'JSR $AD8A'.
Er zijn ook routine's die een geheel getal in een 'FAC-getal' veranderen:
JSR $B395: Deze routine verandert een 16-bits getal in het A- en Y-register in een reeel getal. De high-byte van dat getal zit in de A en de low-byte in Y.
JSR $B3A2: *Deze routine verandert een byte-waarde (0-255), die in het Y-register zit, in een reeel getal.*
Als de waarde eenmaal in FAC zit kun je hem voor verschillende doeleinden gebruiken. Je kunt met de verschillende FAC-routine's gaan werken om op die manier ingewikkelde berekeningen uit te voeren. Je kunt ook bepaalde Basic-routine's gebruiken. Hoe ziet b.v. een programma eruit, dat de sinus berekent, in machinetaal? (zie listing 3)
De instruktie is: SYS 49152, waarde. Eerst wordt de waarde binnengehaald met 'JSR $AD9E'. De numerieke waarde wordt dan in FAC geladen. JSR$E26B' is de routine van de Basic-funktie SIN. De waarde waarvan de sinus genomen moet worden moet dan in FAC zitten. 'JSR $BDDD' verandert FAC in een ASCII-string, waarvan het beginadres $0100 is. 'JSR $AB1E' zorgt ervoor dat de string afgedrukt wordt op het beeldscherm.

```
LISTING 3
C000 20 FD  AE JSR $AEFD ;staat er een komma?
C003 20 9E  AD JSR $AD9E ;haalt parameter.
C006 20 6B  E2 JSR $E26B ;Basic-functie SIN.
C009 20 DD  BD JSR $BDDD;verandert FAC in ASCII-string.
C00C 20 1E  AB JSR $AB1E ;string printen.
C00F 60          RTS
```

Verwerken van een string.

Behalve getallen kunnen ook string's verwerkt worden. Ook nu zijn er verschillende manieren om dat te doen.
De instruktie is :
SYS49152,*x*,*y*, "*string*".
Dit is de variant op de Print-At routine.

```
LISTING 4
C000 20 FD  AE JSR $AEFD;komma?
C003 20 9E  B7 JSR $B79E ;haalt waarde.
C006 86 FB     STX $FB
C008 20 FD  AE JSR $AEFD;volgende komma.
C00B 20 9E  B7 JSR $B79E ;volgende waarde.
C00E 86 FC     STX $FC
C010 A6 FC     LDX $FC    ;x-en y-coordinaat voor PLOT-routine.
C012 A4 FB     LDY $FB
C014 18        CLC
C015 20 F0  FF JSR $FFF0 ;Kernal PLOT-routine.
C018 20 FD  AE JSR $AEFD;volgende komma.
C01B 20 A4  AA JSR $AAA4;Basic-instruktie PRINT.
C01E 60          RTS
```

Nadat de x- en y- coördinaat d.m.v. de PLOT-routine zijn ingesteld, wordt er net gedaan alsof de PRINT-opdracht gebruikt wordt (JSR $AAA4). Behalve 'JSR $AD9E' kan ook de CHEGET-routine gebruikt worden om een teken binnen te halen. Listing 5 gebruikt 'JSR $AD9E' en Listing 6 de CHRGET-routine.
De instruktie is:
SYS 49152,kleur, "string".
Adres $0286 (646) bepaalt de kleur van de tekens. Nadat de string met

```
LISTING 5
C000 20 FD AE JSR $AEFD;komma?
C003 20 9E B7 JSR $B79E;haalt waarde.
C006 8E 86 02 STX $0286 ;letterkleur#
C009 20 FD AE JSR $AEFD;volgende komma.
C00C 20 9E AD JSR $AD9E;haalt string.
C00F 20 21 AB JSR $AB21;string printen.
```

```
LISTING 6
C000 20 FD AE JSR $AEFD;komma?
C003 20 79 00 JSR $0079 ;CHRGOT-routine.
C006 85 FB       STA $FB  ;letter naar 251.
C008 20 73 00 JSR $0073 ;CHRGET-routine.
C00B 20 FD AE JSR $AEFD ;volgende komma.
C00E 20 9E B7 JSR $B79E ;haalt waarde.
C011 A5 FB       LDA $FB  ;haalt letter.
C013 20 D2 FF JSR $FFD2 ;Kernal CHROUT-rout.
C016 CA          DEX      ;verminder waarde.
C017 D0 F8       BNE $C011
C019 60          RTS
```

'JSR $AD9E' wordt binnengehaald gebruik je 'JSR $AB21' om de string te printen. Oplettende lezers hebben wellicht gemerkt dat in het Sinusprogramma (listing 3) 'JSR $AB1E' daarvoor gebruikt werd. Dat komt omdat de routine 'JSR $BDDD', die FAC in een string veranderd, het A- en Y-register gebruikt om het beginadres van de string op te slaan. Als we bij $AB1E beginnen worden die 2 registers weer gebruikt om het beginadres in geheugenplaatsen te zetten die de rest van de routine gebruikt om de string te printen. Die geheugenplaatsen zijn $64/$65 (zie uitleg bij $AD9E-routine).
Als we 'JSR $AD9E' echter gebruiken zit het beginadres al in $64/ $65, dus de omzettingsroutine is helemaal niet nodig (het mag zelfs niet gebruikt worden anders gebeuren er rare dingen). We gaan dus verder met 'JSR SAB21' en de string wordt afgedrukt op het beeldscherm in de gegeven kleur.
Dan nu het gebruik van de CHRGET-routine, waarbij de instruktie is:
SYS 49152, letter, aantal keer afdrukken.

De routine op $AEFD doet, behalve controleren of er een komma staat, nog iets: d.m.v. 'JSR $0073' haal je het teken binnen, dat na de komma staat. Dat is in ons geval een letter. Met 'JSR $0079' wordt die laatste binnengehaalde letter in het A-register gebracht en daarna bewaard in adres $FB (251). Waarom wordt dan daarna de CHRGET-routine aangeroepen?
Dat zit zo: we hebben dan wel een letter binnengehaald, maar de Interpreter weet nu niet wat hij ermee aanmoet. Normaal zouden we dan ook een 'SYNTAX ERROR' krijgen.
Door echter de CHRGET-routine aan te roepen halen we nu de komma binnen (we slaan de letter als het ware over) die daarna meteen door de 'JSR $AEFD' gecontroleerd wordt.
Vervolgens wordt die letter net zolang afgedrukt tot de gegeven waarde d.m.v. 'DEX' nul heeft bereikt.
Tot slot nog het gebruik van de routine op $E1D4. Zoals je al gelezen hebt wordt die gebruikt om de filenaam en het devicenummer van de 'LOAD'-,'SAVE'-en 'VERIFY'-opdracht te halen. Het laatste programma demonstreert hoe je het beeldscherm kunt saven. Type maar het hele beeldscherm vol, type dan 'SYS 49152, "filenaam",8' (of '1'voor cassette), haal het scherm weg en type 'LOAD "filenaam",8,1' (of '1,1' voor de cassette).

Rene Janssen ●

```
LISTING 7
C000 20 FD AE JSR $AEFD
C003 20 D4 E1 JSR $E1D4
C006 A2 E7       LDX $E7
C008 A0 07       LDY $07
C00A A9 00       LDA $00
C00C 85 FB       STA $FB
C00E A9 04       LDA $04
C010 85 FC       STA $FC
C012 A9 01       LDA $01
C014 85 B9       STA $B9
C016 A9 FB       LDA $FB
C018 4C D8 FF JMP $FFD8
```

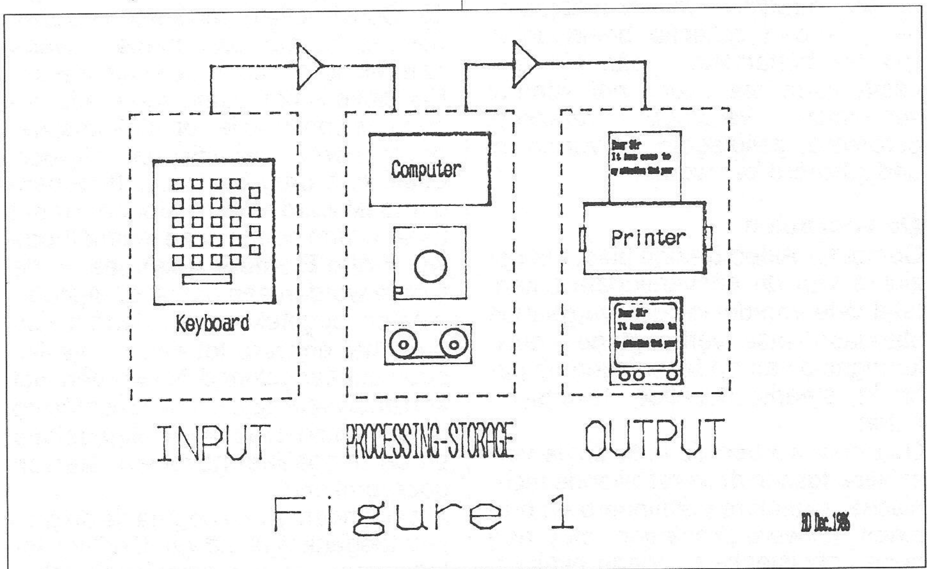
Computer Aided Design is niet meer uit de moderne architectuur en industrie weg te denken. Van de bouwtekeningen voor uw zomerhuisje, electronische circuits voor audio-apparatuur, lenzenstelsels voor camera's tot spaceshuttles, alles rolt tegenwoordig uit de door CAD-software gestuurde computers. Vroeger was CAD een toepassing voor grote systemen. De laatste tijd is de CAD-techniek echter in een stroomversnelling geraakt en verschijnen steeds meer systemen voor de kleine kantoor- en microcomputergebruiker. Ook voor de Commodore C-128 (en natuurlijk de Amiga en CBM PC's) zijn inmiddels een aantal CAD-pakketten beschikbaar gekomen.

CAD op de C-128

Een kennismaking met Computer Aided Design

CAD ging van mainframe naar minicomputer en van minicomputer naar microcomputer. Een C-128 is uiteraard iets anders dan een PC. CAD op de Commodore kan daarom nooit zo uitgebreid zijn als op een groter systeem, maar als introductie tot de CAD-techniek is een Commodore-huiscomputer zo gek nog niet.

Wij zien CAD op de Commodore C-64 en C-128 daarom als leerpakketten die de hobbyist en student de mogelijkheden van CAD laat zien. Bovendien zijn een aantal tekeningen op kleinere schaal, bijvoorbeeld de bouwplannen voor een schuurtje of een enkel mechanisch onderdeel, best bruikbaar. Huiskamer-CAD slaat daarbij naast kantoor-CAD zeker geen slecht figuur.



Met CAD-software getekend schema

Wat is CAD?

Achter het begrip Computer Aided Design schuilt veel meer dan alleen het maken van technische tekeningen. Na installatie gaat voor de gebruiker een geheel nieuwe wereld open van grafisch ontwerpen, illustraties (art work, dia's, overheadsheets) stroomdiagrammen, bedradingsschema's, layout van formulieren, bouwtekeningen, spreadsheet en tekstverwerkings-toepassingen.
Betrekkelijk nieuw voor de Amiga zijn de simulaties aan de hand van tekeningen op een CAD-systeem, die er voor zorgen dat een klant op de monitor kan bekijken hoe zijn toekomstige bungalow of fabriek er uit komt te zien. Deze mogelijkheden hangen uiteraard van de hard- en softwareconfiguraties af, maar zelfs met een eenvoudig Commodore XT-tje of een C-128 en een goedkoop CAD-pakket kan menigeen al uit de

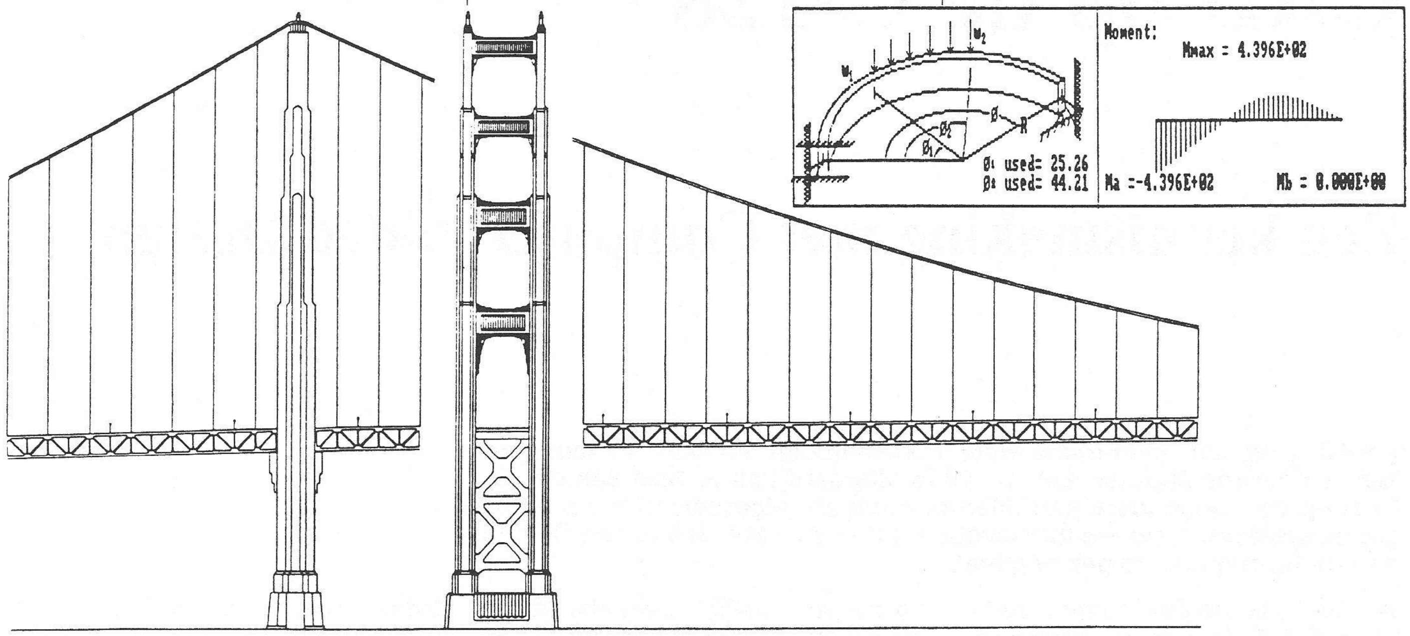
voeten. De popularising van de PC en de huiscomputer heeft CAD in haar kielzog meegezogen. Werd CAD in het verleden door specialisten op een mini- of mainframe-computer beoefend, tegenwoordig is er geen sprake meer van eiland-vorming, want iedereen kan zelfstandig met de eigen PC aan de slag.
CAD is in feite niet meer dan met behulp van de computer architectonische en industriële ontwerpen maken op het beeldscherm. Alhoewel de computer een deel van de werkzaamheden van de ontwerper overneemt, kan niet iedereen zomaar de prachtigste ontwerpen maken, want met CAD ben je nog geen geschoold ontwerper. CAD-pakketten helpen de (on)ervaren gebruiker gelukkig wel op tal van manieren. Er is altijd sprake van een of ander **coördinaten-systeem** dat de juiste afmetingen en dimensies (automatisch) vastlegt. Een schaalverdeling helpt bij het bepalen van de juiste maten, zodat de tekening later qua schaal exact met de werkelijke afmetingen overeenkomt.
Ook als je met een potlood nog geen rechte lijn op papier kunt krijgen, valt er met eenvoudige CAD- toepassingen nog wel iets te beginnen. De software **tekent** lijnen en standaardfiguren of -vormen **automatisch** na het invoeren van de bijbehorende coördinaten. Lastige veelhoeken, circels, bochten, en loodrechte hoeken behoeven dus geen tekenprobleem meer te zijn. Hetzelfde geldt voor het aanbrengen van Fill-patronen, schaduweffecten en het simuleren van 3D.
Vele CAD-pakketten beschikken zelf over complete **symbool**- en **figuurbibliotheken** met huisje, boompje, beestje en vele standaard industriële vormen. Ook zijn er dikwijls verschillende **fonts** ingebouwd voor het op attractieve wijze verduidelijken van de verschillende onderdelen van het ontwerp.
Zoom en scrollcommando's maken het mogelijk om een veel groter ontwerp te maken dan op het monitorscherm past en details voor nauwkeurige bestudering en correctie uit te vergroten.
CAD-programma's zijn **'objected-based'**. Dat wil zeggen dat de software de verschillende tekenobjecten niet als een collectie beeldpunten (pixels) behandelt, maar als een vaste vorm die over het scherm verplaatst, vergroot/ verkleind, geroteerd, gespiegeld, vervormd, of gedupliceerd kan worden.

De voordelen

Computer Aided Design biedt ten opzichte van de conventionele tekentafel vele voordelen. Een vergaande standaardisatie verhoogt de nauwkeurigheid van de tekeningen en gebruikt steeds dezelfde tekensymbolen.
Daarmee worden de individuele verschillen tussen de verschillende technische tekenaars verminderd en ontstaan ontwerptekeningen met een hoge consistentie en goede kwaliteit. Veranderingen zijn snel en betrekkelijk eenvoudig door te voeren. Gewoon de oude tekening inladen, de wijzigingen aanbrengen en een nieuw up-to-date ontwerp rolt uit de plotter. Kortom, CAD staat borg voor een snellere en efficiëntere bedrijfsvoering.

CAD-toepassingen kunnen in veel gevallen de veiligheid aanmerkelijk vergroten.

In de industrie wordt CAD veel gebruikt voor het electronische testen van het gemaakte ontwerp. Simulatie van stress en andere ontwerp-onvriendelijke factoren zoals hitte, druk, wind en golven maken een tijdige aanpassing van het ontwerp mogelijk. Constructiefouten komen nu zonder het gevaar voor mensenlevens aan het licht. Een andere aantrekkelijke optie is het testen van reeds bestaande constructies op toekomstige, onvoorziene, belastingen. Bijvoorbeeld een brug die voor 10-tonners gemaakt werd en nu 20-tonners moet torsen. Men noemt deze testmethode wel **Finite Element Analyses**. In de ruimte worden een aantal draagcoördinaten opgetekend en daarna volgens het ontwerp tot een ruimtelijke structuur aangekleed. Nu worden met een analyseproggramma krachten op deze constructiepunten uitgeoefend en de (mogelijke) gevolgen daarvan doorgerekend.
Tot slot noemen wij nogmaals de grotere toegankelijkheid van CAD-systemen voor de niet-technische tekenaar.



CAD-toepassingen bestrijken het gehele gebied tussen schematisch 'overall-beeld' en gedetailleerd precisiewerk.

De basisconfiguratie

Een C-128 computer met bijbehorende software alleen is niet voldoende voor het bouwen van een CAD-configuratie. Daar komt heel wat meer bij kijken. In het algemeen bestaat een thuis- of school-CAD-systeem uit de volgende componenten:

- Een **C-128** met floppy disk en voldoende vrij RAM. Voor gedetailleerde ontwerpen is een RAM-uitbreiding tot 256K wenselijk. Binnenkort komt ook een harde schijf voor de C-128 en C-64 beschikbaar en die kan heel wat CAD-data aan.
- Een goede **HI-RES monitor** is vrijwel onmisbaar. Er is keuze uit monochroom- of kleurentypen. Een kleurenmonitot is aanzienlijk duurder, maar maakt het beeld door kleurschakeringen en kontrasten er wel duidelijker en overzichterlijker op.
- Een **CAD-pakket** dat aan de eigen eisen voldoet. Niet iedereen gebruikt CAD voor hetzelfde doel en in vele gevallen gaat het zelfs om meerdere toepassingen. Net als elke andere software-aankoop is het van groot belang om een prioriteitenlijstje te maken. Aan toeters en bellen heeft de gemiddelde gebruiker niet zoveel. Wel aan een gebruiksvriendelijk pakket dat precies doet wat het moet doen, en nog redelijk geprijsd is ook. Het hier

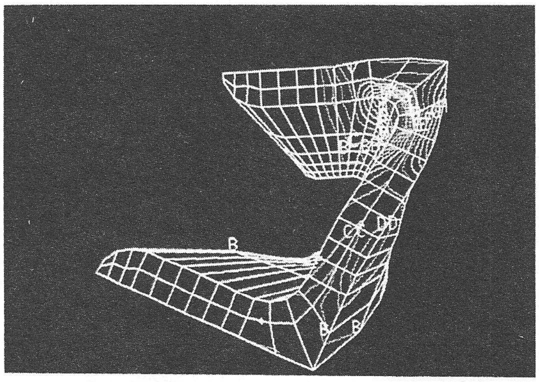
CADPAK-128

Het Britse softwarehuis Adamsoft heeft onlangs het HI-RES ontwerp- en tekenpakket **CADPAK-128** op de markt gebracht. Dit disk-based CAD-pakket tekent 40-koloms grafische beelden op de Commodore monitor en biedt printerdrivers voor 640 x 360 dots hardcopies. De gebruiker heeft de keuze uit twee displayscreens:

° Het **Main Screen** van 640 x 360 pixels. De werkelijke resolutie in deze display-modus bedraagt 320 x 200 pixels, maar de gebruiker heeft gewoon een twee maal zo groot schermoppervlak tot zijn/haar beschikking.

° Het **gewone scherm** van 320 x 200 pixels.

Via het screen window (beeldvenster) kunnen objecten



vrijelijk tussen de beide displays gekopieerd worden. Het tekenen zelf kan met de cursortoetsen, een muis of een lichtpen. De lightpen-optie is zeker de moeite van het overwegen waard daar het gebruikersgemak, snelheid en de mogelijkheden van de software er aanzienlijk mee vooruitgaan. Het aansluiten van de pen ging vrijwel probleemloos, gewoon een kwestie van inpluggen, evenals het instellen met behulp van het lightpen-menu.

CADPAK-128 biedt de gebruiker de volgende mogelijkheden:

° **Overlay en printscaling.** De grootte van de hardcopy kan door de gebruiker vrijelijk ingesteld worden. Via overlays zijn behoorlijk gecompliceerde constructietekeningen mogelijk.

° Volledige **menusturing**.

° **Saven** van gemaakte tekeningen in **BASIC** en **6502 machinetaal**.

° **Exact scaled output**. De uitgedraaide tekening is precies op schaal. Naar keuze in (centi) meters, kilometers en inches. De schaal wordt tijdens het initialisatieproces vastgelegd. Verder draagt deze optie er zorg voor dat cirkels ook als cirkels en niet als eieren afgedrukt worden.

° Het **object management systeem** (OMS) hanteert beelden van 16 x 16 pixels op het normale scherm. De gebruiker kan deze objects voor latere toepassingen saven. Er kunnen maximaal 104 OMS-objects tegelijk gebruikt worden. Onder het OMS vallen ook enkele standaard-bibliotheken op de systeemschijf met onder andere kaarten, meetkundige figuren en electronische symbolen.

° **Templates** zijn lijntekeningen die voor later gebruik gesaved kunnen worden. Zo kunt u de bibliotheek zelf verder uitbreiden. Het bijzondere van de templates is dat zij weer in andere tekeningen ingebouwd kunnen worden. Templates zijn vanuit BASIC aan te roepen en zo ook in zelfgemaakte programma's in te bouwen.

° **dupliceervenster**. Via het scherm-venster (window) kan de ontwerper een object kopiëren, roteren of spiegelen en daarbij ook nog in grootte wijzigen.

° Drie **Fonts** met tekst in vier verschillende grootten en 3D-letters. Met de **object editor** kunnen desgewenst nieuwe Fonts ontworpen worden.

° Alle gebruikelijke **standaard tekentechnieken**.

° Zeven verschillende **fill-patronen** bestaande uit 8 x 8 pixels.

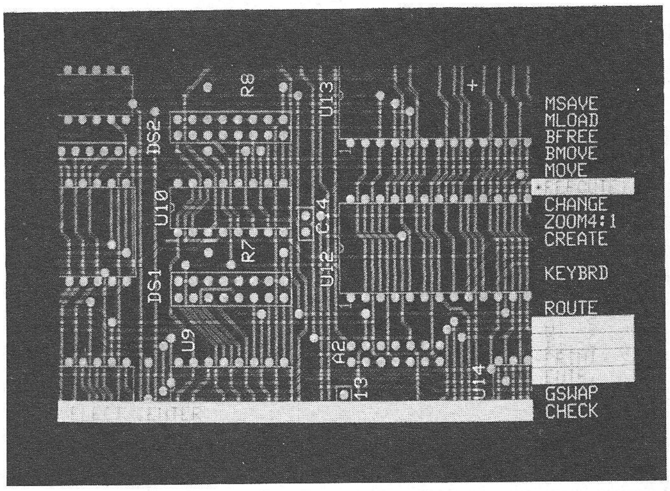
° De **Try Again**-functie die het laatst gegeven commando ongedaan maakt.

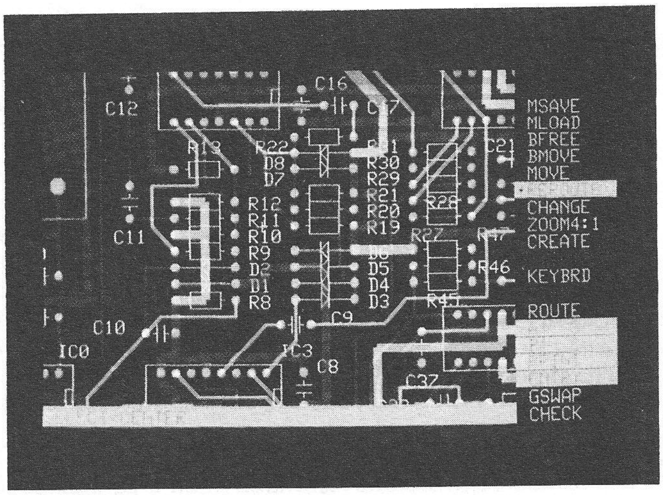
° Een 86-pagina's tellende duidelijke **Engelse gebruiksaanwijzing** met tal van lessen en voorbeelden.

° **Full printer support** voor Commodore seriële en parallel Centronics printers. Helaas worden er maar enkele plotters ondersteund! Eén van de weinige nadelen van dit CAD-pakket.

CAD op de C-128 is nu voor iedereen bereikbaar. Verwacht echter geen professionele kwaliteit, maar wel bruikbare en vooral leerzame resultaten. En om de prijs hoeft u het echt niet te laten.

CADPAK-128 kost rond de *f* 125,-. Daar komt de prijs van de lightpen nog bij.





Een andere CAD-toepassing is het tekenen van electronische schema's.

besproken **CADPAK-128** voldoet aan de meeste van deze eisen.

- **Input-devices**. Het tekenen met een CAD-pakket kan geschieden met tal van invoerapparatuur of *"stuurwijzers"*. Voor serieus gebruik zijn de cursortoetsen van het keyboard eigenlijk totaal ongeschikt. Zelfs als tijdelijke noodoplossing. Vrijwel elk CAD-pakket werkt met een **muis**. Helaas geven deze digitale knagers nogal eens last. Onbetrouwbare overbrenging, slechte compatibiliteit met hard- en software en het crashen van het systeem bij het per ongeluk uitvoeren van een verboden manoevre vormen slechts een kleine greep uit de vele gruwelen waarmee de beginnende CAD-man/vrouw geconfronteerd kan worden. Dat hoeft echter niet als men van meet af aan met een goede C-128 compatibele muis en stuursoftware start.
Behalve muizen is er wat de stuurwijzers betreft nog keuze uit lichtpennen, graphic tablets, joysticks en track/joy-ballen. Elk device heeft zo zijn eigen enthousiaste aanhangers/boe-roepers, voor- en nadelen, en (on)mogelijkheden. Het is moeilijk om hierin verantwoord te adviseren. Dikwijls vindt u in het manual de nodige aanbevelingen en afraders die een verstandige keuze vergemakkelijken.

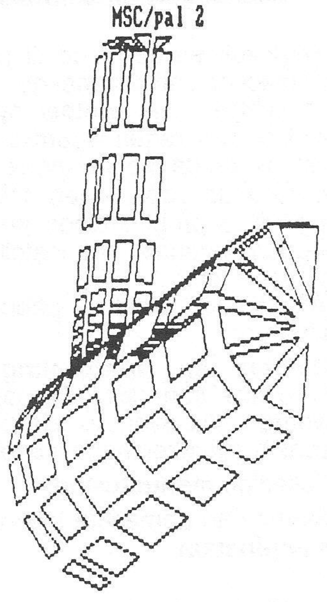
- Enkele CAD-pakketten ondersteunen **scanners** en **digitizers**. Daarmee kunnen al bestaande ontwerpen in het CAD-systeem ingevoerd en gewijzigd worden. Een nuttig hulpmiddel voor de bouwkundig- of industrieel-tekenaar die nog een kast vol oude ontwerpen had liggen of het reklamebureau dat bestaande afbeeldingen in een nieuwe campagne wil integreren. Ook voor de C-128 en C-64 zijn met name in Duitsland diverse scanners en digitizers met tal van artistieke en CAD-mogelijkheden te koop.

- Voor het maken van perfecte hardcopies komen zowel plotters, printers als beeldrecorders in aanmerking. Van oudsher is de **plotter** het geëigende tekeninstrument voor CAD/CAM-toepassingen. Een goede plotter kan een zeer hoge grafische resolutie met een perfecte afdrukkwaliteit bereiken. Voor een plotter geldt in feite hetzelfde als voor de monitor. Een inferieur model haalt niet uit het systeem wat er in zit. Behalve papieren hardcopies kunnen de meeste plotters ook transparante folies (overheadsheets) beschrijven. Voor de Commodore is er keuze uit de, eigenlijk niet serieus te nemen, op papierrol schrijvende 1520 en aanverwante plotters waarvan het papierformaat veel te klein is en de redelijk geprijsde *Comx PL-80*.

Wie niet zo aan de optimale kwaliteit tilt en even snel een redelijk tot goed afdrukje wil uitdraaien is bij een **24-naalds kleuren-matrixprinter** of een goede **inkjet-printer** aan het juiste adres. Een alternatieve keuze is bijvoorbeeld de nieuwe Commodore MPS 2000 C kleurenprinter. De moderne printers zijn grafisch heel wat mans, maar kunnen nog niet echt aan de plotter tippen.

- **Koppeling aan een database**. Met behulp van een database kan een schat aan grafische en alfanumerieke gegevens aan het CAD-ontwerp gekoppeld worden. De dienst publieke werken van een gemeente kan zo bijvoorbeeld het rioleringssysteem in kaart brengen en bij alle buizen de diameter, diepte, gebruikte materialen, ouderdom, onderhoudsgevoeligheid, de bij onderhoud af te sluiten gedeeltes e.d. vermelden.

De mogelijkheid nu ook op de Commodore-machines CAD-toepassingen te laten draaien, betekent opnieuw een grote technische verbetering. Zoals eigenlijk het afgelopen jaar al vaak is voorspeld, blijkt in de praktijk de hobby-sector en de professionele markt inderdaad steeds dichter naar elkaar toe te groeien. ●



HET ONMOGELIJKE BELASTINGPROGRAMMA

door Rob Bakker

Zelfs een Minister van Financiën heeft wel eens een zwak moment. Op een avond, nadat de spruitjes met komkommer en radijsjes slecht waren gevallen, besloot hij dat de belastingaangifte van iedere burger vereenvoudigd kon worden. Dat sloeg in.
Om de zaken voortvarend aan te pakken werd er een commissie opgericht die onderzocht hoe de belastingaangifte vereenvoudigd diende te worden. De snelheid bleef erin, dus reeds na een jaar kon de commissie de oplossing al rapporteren: de belastingaangifte moest gecomputeriseerd worden. Geef iedere belastingplichtige een Belasting- programma waarmee hij fluitend zijn formulier kan invullen. Dat programma moest het mogelijk maken dat iedereen precies snapte hoe het biljet ingevuld moest worden. Dat sloeg nog meer in.
Om de snelheid erin te blijven houden werd er een prijsvraag uitgeschreven voor verkiezing van Bouwmeester van het Belastingprogramma. Iedereen wist dat het natuurlijk onmogelijk was om de belastingaangifte te vereenvoudigen, maar alle programmeurs van naam dongen toch mee naar de titel. Behalve de onsterfelijkheid en een dikke kans op een Nobelprijs, zat er aan die opdracht ook een leuk bedrag verbonden van een paar miljoen. Een zwak moment van een Minister van Financiën staat immers gelijk aan een bui van hysterisch geld uitgeven bij iedere andere minister.
De top-ontwerpers die zich normaal bezighielden met ordinaire zaken als het SDI - Star Wars - computer-programma zagen de nieuwe uitdaging en lieten de ruimte verder over aan de Russen.

Rob Bakker is schrijver van boeken en verhalen. Voor Commodore Info maakt hij een serie korte verhalen, waarbij de computer in een humoristisch, maar soms meedogenloos daglicht wordt geplaatst.

Iedereen begreep dat hier de kans lag om in alle geschiedenisboekjes te komen. En wat was er mooier dan de gehele mensheid te helpen met een Begrijpelijk Belastingprogramma? De uitvinding van de penicilline was leuk, maar genezen van onbegrijpelijke aanslagen is nog veel mooier.

Uit de vele duizenden inzenders werd er met grote snelheid reeds binnen het jaar eentje uitverkoren, wiens ontwerp de jury voldoende aansprak. Het was een jongentje van zestien jaar, dat nog nooit eerder een belastingbiljet had hoeven invullen. Zodoende had hij de verfrissende creativiteit gehad om de problematiek ongedwongen te benaderen. De blik van iedere andere goedbetaalde programmeur was immers zo vertroebeld door jarenlang zoeken naar constructies om de belastingen te ontwijken, dat ze de materie alleen nog maar vanuit het meest ingewikkelde standpunt konden benaderen.
Het uitverkoren ventje kreeg een eigen kantoortje en per dag twee bonnen voor de kantine. Na een maand waren drie printers doorgebrand, en vier computers geheel waanzinnig geworden. Intel wachtte ongeduldig met het uitbrengen van Chip 486 om die alsnog aan de eisen van het nieuwe belastingprogramma aan te passen. Het was duidelijk dat daarmee de Japanners een dodelijke slag toegebracht kon worden in de internationale concurrentie.
Na een maand en drie dagen kon het ventje het Belastingprogramma presenteren. (De laatste drie dagen waren nodig om de handleiding van 1004 pagina's terug te brengen tot een velletje A4. Exclusief garantiebepalingen. Het was echt een heel slim ventje.)
Hij werd welwillend ontvangen door de Minister van Financiën, die hem een glas coke presenteerde (verantwoord onder Post Representatie 24 B

art. 3, tweede lid, boeknummer 725387) en die vervolgens het programma doornam.
'Tja', zei de Minister na twee minuten (het was een kort progamma). 'Het is inderdaad gelukt.' Maar de Minister keek er erg ongelukkig bij. 'Weet je', zei hij vertrouwelijk tegen de knul,'dat onze defensie-inspanningen ieder jaar 3 procent stijgen? De Rus, begrijp je wel?'
Het financieren van een beetje oorlog kost al gauw heel wat, begreep de jeugdige programmeur.
'En de kosten voor sociale zaken: gestegen met maar liefst negen procent', riep de Minister.
Medemenselijkheid kost inderdaad niet niets, snapte de jongen.
'En dan de post Representatie: gestegen met 85,34678%!' gilde de Minister. Iedereen weet dat zoiets onvermijdelijk is.
De Minister ging verder. 'Als ik nu naar vraag 3b kijk van ons Eenvoudige belastingprogramma, dan is die nu wel erg duidelijk geworden...' De Minister pauzeerde onheilspellend.
'Vroeger verdienden we alleen al op vraag 3b 184 miljoen. Maar nu...', de stem werd schel,' is die vraag zo duidelijk geworden dat iedereen weet hoe die hem goed moet invullen en dat gaat ons 340 miljoen 178 duizend, 64 gulden en 36 cent aan belastinginkomsten schelen.'
De Minister keek radeloos. 'Hier vraag 18: kochten we vroeger twee Walrussen voor. Nu gaat ons dat 530 miljoen, 194 duizend, 32 gulden en 54 cent kosten.' De Minister was bleek weggetrokken.
'Geen mens zal nog uit onwetendheid teveel geld gaan betalen en het gevolg is dat de belastingen omhoog moeten en dat wil toch niemand? Hoe moet het anders met de Rus, onze medemenselijkheid en erger nog: onze representatie?' Driftig veegde de minister een glas van tafel (Post Rampenplan, 1 ste artikel, lid 3, boekno. 007).
'We praten er niet meer over', zei de Minister, stak de floppie met het programma in zijn zak (je kon nooit weten voor privégebruik) en stuurde de jongen weg. 'U krijgt uw honorarium nog wel', riep hij vals lachend aan de deur.
Het ventje kreeg op papier zijn vijf miljoen gulden salaris, tegelijk met een aanslag over dat bedrag. Na invulling van de niet-vereenvoudigde vragen, bleek hij nog drie jaar lang een krantenwijk te moeten nemen om de aanslag van 101% over zijn verdiende honorarium (reeds ingehouden volgens artikel 36) te kunnen voldoen. ●

Basic Min

Een rubriek van Nico Baaijens me

Het heeft klachten geregend de laatste weken over de BASIC- miniatuurtjesrubriek. Niet over de inhoud of over de kwaliteit, maar over het feit dat deze, kennelijk zeer geliefde rubriek twee keer niet in de Commodore Info heeft gestaan. Laten we hopen dat de PTT niet nogmaals de domme fout begaat om mijn diskette met miniatuurtjes-kopij zoek te maken.

In de stapel brieven van lezers/programmeurs zaten er enkele, die meenden dat de rubriek niet is verschenen wegens gebrek aan inzendingen. Deze mensen zijn onmiddellijk in het toetsenbord geklommen en hebben miniaturen ingezonden. Laat dit goede voorbeeld goet doen volgen, hoewel we tot nu toe gelukkig geen gebrek aan kopij hebben. Ook nu weer moet ik met mijn twee Info-paginaatjes woekeren.

Schermeffecten

Leo M.H. Delhey uit Bladel houdt ervan om de aandacht te trekken via het beeldscherm. Hij ontwikkelde een horizontale en vertikale schermvuller, die als echte eyecatchers bruikbaar zullen zijn.

```
10  A=1524:F=0
20  B=0:C=0:D=0:K=81
30  POKEA,K
40  B=B+1:C=C-1
50  D=A+B:E=A+C
60  K=K+6:IFK87THENK=81
70  POKED,K:POKEE,K
80  IFB=500THEN100
90  GOTO40
100 PRINT"hft Clr/Home"
```

```
10  X=0:C=0:F=0
20  FORY=0TO480STEP40
30  GOSUB150
40  Y=-Y
50  GOSUB150
60  Y=-Y
70  NEXTY
80  IFX=20THEN180
90  C=C+1
100 IFC=3THENC=1
110 IFXTHENX=-X:GOTO130
120 IFC=2THENX=-X:GOTO20
130 X=X+1
140 GOTO20
150 POKE1523+Y+X,42
160 IFK=16THENK=0
170 RETURN
```

Erg aardig vond ik het titelscherm van 'Grovo Vision Rotterdam'. De inzending is van David de Vos (16) uit Rotterdam, die een blits titelscherm samenstelt met van boven naar beneden vallende letters. Probeer het eens:

```
10  DIMU(100)
20  POKE53280,0:POKE53281,0
30  PRINTCHR$(147)
31  PRINTTAB(8)"ZEVEN MAAL CRSR DWN"
32  FORT=0TO7:SYS59765:FORR=0TO250:
    NEXT:NEXT40FORK=8TO29
50  READU(K)
60  FORT=0TO6
70  POKE1024+K+T*40,U(K)
80  FORN=0TO50:NEXT
90  POKE1024+K+T*40,32
100 NEXT
110 POKE1024+K+T*40,U(K)
120 NEXT
130 DATA7,18,15,22,15,45,22,9,19,9,15,14
140 DATA45,18,15,20,20,5,18,4,1,13
```

Screenfill Special is het beeldschermeffect van Bas Ording uit Blaricum:

```
10  A=0:B=0:C=54272:D=160
20  FORT=1024+A+BTO1063-A+BSTEP1
30  POKET,D:POKET+C,RND(1)*255:NEXT
40  FORT=1103-A+BTO1983-A-BSTEP40
50  POKET,D:POKET+C,RND(1)*255:NEXT
60  FORT=2023-A-BTO1984+A-BSTEP-1
70  POKET,D:POKET+C,RND(1)*255:NEXT
80  FORT=1944+A-BTO1064+A+BSTEP-40
90  POKET,D:POKET+C,RND(1)*255:NEXT
100 IFATHENA=A+1:B=B+40:GOTO20
```

En dan hebben we ook nog een deinend beeld van E.A. van den Bos uit Baarn:

```
10  FORX=0TO76:READA:POKE50000+X,A:NEXT
20  SYS50000
30  DATA120,169,198,133,251,165,251,141,
    22,208,198,253,208
40  DATA6,169,10,133,253,198,251,165,251,
    201,191,240,13
50  DATA230,252,230,252,173,18,208,197,252,
    208,249,240
60  DATA222,169,193,133,251,165,251,141,22,
    208,198,253,208
70  DATA6,169,10,133,253,230,251,165,251,201,
    200,240,193
80  DATA230,252,230,252,173,18,208,197,252,
    208,249,240,222
```

Games

'Met dit spelletje is het onmogelijk om te winnen', schrijft Laurens van Kooy (16) uit Straatsburg in Frankrijk. Het is een zeer compact geschreven NIM-variant met als spelregel dat wie de laatste lucifer pakt, heeft verloren. Onmogelijk om te winnen is dit spel niet, maar wel erg lastig.

```
1   PRINT"hft Clr/HomeWie de laatste pakt heeft
    verloren":S=21
2   FORX=1TOS:PRINT"I";:NEXT:PRINTTAB(21)"ER
    ZIJN ER NOG"S"
3   INPUT"HOEVEEL (1,2 OF 3)";A:C=4-A:IFAINT(A)
    ORAORA3THEN3
4   PRINTTAB(22)"IK PAK
```

iatuurtjes

korte tot zeer korte programma's.

```
    ER"C:S=S-4:IFS=1THENPRINT"I";TAB(22)"ER
    IS ER NOG 1":GOTO6
5   GOTO2
6   PRINT"DE LAATSTE IS VOOR JOU, IK HEB
    GEWONNEN!"
```

Blijven we nog even bij de games dan is ook het kleintje van Rene Lergner uit Papendrecht aardig. Men besture de slang met de 3 en de 9 en men probere alle A's te ontwijken. Na drie keer proberen kwam ik tot een score van 864. Wie verbetert dat?

```
10  POKE1444+X,0
20  GET A:X=X+(A=3)-(A=9)
30  PRINT:POKE 1984+RND(1)*40,1
40  IF PEEK(1444+X)1 THEN I=I+1:GOTO 10
50  PRINT"SCORE:";I
```

Utility's

Tot de nuttige (hulp)programmaatjes reken ik de bijdrage van L. Hommel uit Tilburg. De IBM PC maakt het mogelijk om de inhoud van het (grafische) beeldscherm met de toetsencombinatie Shift/PrtSc naar de (grafische) printer te dumpen. Met de C64 kan dat ook, alleen moet dan wel even onderstaand machinetaal-miniatuurtje worden geRUNd, waarna SYS50000 hetzelfde doet.

```
40  FORX=0TO94:READA:POKE50000+X,A:NEXTX
50  DATA169,4,133,186,169,126,133,184,169,
    0,160,4,133,113,132,114,133
60  DATA183,133,185,32,192,255,166,184,32,
    201,255,162,25,169,13,32,210
70  DATA255,32,225,255,240,46,160,0,177,113,
    133,103,41,63,6,103,36,103
80  DATA16,2,9,128,112,2,9,64,32,210,255,200,
    192,40,208,230,152,24,101
90  DATA113,133,113,144,2,230,114,202,208,
    205,169,13,32,210,255,32,204
100 DATA255,162,126,76,195,255,96
```

Een aardige utility kwam van D. de Vos uit Rotterdam. Run onderstaand miniatuurtje. Het wordt dan mogelijk om met een machinetaal-monitor de ROM-commando's te veranderen. Deze commando's staan op de adressen A0000-C0000. Verlaat de monitor vervolgens en type: POKE 1,54.

```
80  FORT=49152TO49176:READA:POKET,A:NEXT
90  DATA160,159,200,140,11,192,140,14,192,
    189,0,160,157,0,160,232,224
100 DATA255,208,245,192,191,208,234,96
110 SYS 49152
```

Commodore 16

De vergeten groep van de C16-gebruikers komt toch weer aan bod. Van Raimond Wolters uit Terneuzen een grafisch miniatuurtje om een cirkel op te bouwen uit rechthoeken.

```
10  COLOR 0,1
20  COLOR 1,2
30  GRAPHIC 2,1
40  A=RND(1)*20+10
50  FOR L=0 TO 300 STEP 3
60  BOX 1,100,30,220,130,L
70  NEXT L
80  FOR L=1 TO 2000:NEXT L
90  GRAPHIC 0,1
```

A. Stremler uit Papendrecht schrijft een brief aan 'de mensen van C=Info' en doet daar drie miniatuurtjes voor de C16 bij. We laten er twee zien. Het eerste is een beeldschermeffect en het tweede ontlokt muzikale geluiden aan de computer.

```
5   COLOR 0,1:COLOR 1,2
10  GRAPHIC 1,1
20  FOR A=0 TO 100
30  CIRCLE,67,90,A,A,,,,12
40  NEXT A
```

```
5   PRINT CHR$(147)
10  VOL 8
20  FOR X=50 TO 510 STEP 4
30  SOUND 1,X,1
40  SOUND 1,X+23,1
50  NEXT X
60  FOR X=0 TO 120 STEP 4
70  SOUND 1,X,1
80  SOUND 1,X+23,1
90  NEXT X
100 FOR A=0 TO 123
110 FOR X=120 TO 120 STEP 4
120 SOUND 1,X,1
130 SOUND 1,X+23,1
140 NEXT X
150 NEXT A
160 FOR X=100 TO 0 STEP-4
170 SOUND 1,X,1
180 SOUND 1,X+23,1
190 NEXT X
195 FOR A=1 TO 80
200 FOR X=0 TO 0 STEP 4
210 SOUND 1,X,0
220 SOUND 1,X+23,1
230 NEXT X
240 NEXT A
```

Tot slot voor de C16 nog de bijdrage van Paul Wessels uit Den Haag:

```
10  VOL 7
20  T=INT(RND(1)*1000)+1
30  S=INT(RND(1)*150)+100
35  FOR G=1 TO 3 STEP 1
40  SOUND G,T,S
45  NEXT G
50  K=INT(RND(1)*16)+1
60  H=INT(RND(1)*8)+0
70  COLOR 4,K,H
80  COLOR 0,K,H
90  GOTO 10
```

We zetten er weer een punt achter voor deze maand. Obbe Vermeij uit Capelle a/d IJssel zorgt voor de uitsmijter. Zijn miniatuurtje geeft aardige effecten in de border. Probeer daarbij ook de borderkleur te veranderen voor de RUN.

```
10  FOR T=0 TO 8:READ A:POKE
    49152+T,A:NEXT:SYS49152
20  DATA238,32,208,206,32,208,76,0,192
```

Wie zei dat het onmogelijk was om zelf leerprogramma's te maken voor het gebruik in de onderwijspraktijk? Commodore-Info vindt, dat het wel degelijk de moeite loont, als leerkracht zelf aan de slag te gaan. We gaven reeds een aantal handreikingen in de vorm van voorbeeldprogramma's. We gaan er mee door en lossen een belofte in. Deze keer gaat het over het maken van Multiple Choice-testen.

Zelf leerprogramma's maken in BASIC



Deel 5: Meerkeuze testen

Wie kent niet de befaamde multiple choice toetsformulieren van het CITO. Computerkaarten met een aantal hokjes erop. Is het nu antwoord A, B, C of D ? Het potloodpuntje zorgvuldig binnen de lijntjes en invullen maar. Soms gokken, soms in één keer raak. In elk geval een redelijk betrouwbaar en vooral snel medium om kennis te toetsen.

Ook in de klassepraktijk van de basisschool is het wel eens handig, bepaalde leerstof snel te toetsen. Niet om te kijken waar de problemen liggen, maar wel om eens te zien hoe de vlag er bij hangt ten aanzien van de aangeboden en hopelijk verwerkte leerstof. Een soort tussenstand dus. Vooral de factor snelheid heeft hier prioriteit. Daar is een computer immers goed in. En waarom zouden we dat niet benutten.

De Multiple Choice opdracht

Bij een opdracht met meerkeuze antwoord, om de Nederlandse aanduiding maar eens te gebruiken, is de beantwoording van de vraag reeds voorgestructureerd. Er zijn doorgaans vier mogelijke antwoorden, waarvan meestal één antwoord onzin is. De andere drie antwoorden zijn zo geformuleerd, dat vergissen zeer goed mogelijk is, tenzij de leerstof goed beheerst wordt. U kent dat wel. De CITO is met die materie ook zeer in de weer. Met een beetje training kunt u die vragen met bijbehorende antwoorden ook bedenken. Want dat moet u wel zelf doen. Dat kan een computer (gelukkig) niet zelf! Het programma, dat hier is afgedrukt, verzorgt slechts het verwerken van de opdrachten, die u in een eerder stadium al op papier zette.

De programmastructuur

De opzet van het programma is eenvoudig. Geen ingewikkelde IN/UIT handelingen, geen gebruik van extern geheugen (cassette of diskette), zodat iedereen ermee kan werken. Eenvoudig van structuur om het voor een ieder leesbaar te houden en om het toegankelijk te maken. Immers, het is niet het enige echte multiple choice programma, maar een mogelijkheid uit vele.

Alle programma's voor het gebruik in de klassesituatie moeten aan bepaalde eisen voldoen, vinden wij. Daar hebben we het al eens over gehad. Ook deze software, hoe eenvoudig ook, voldoet ons inziens aan die gestelde eisen. Het programma is ruim voorzien van REMarks en de blokjes zijn afgebakend door middel van lege regels met een dubbele punt. Dit zal de leesbaarheid vergroten. Verder staan er tussen alle statements spaties. Die zijn ook voor het leesbaar houden van de regels, maar mogen door de meer ervaren programmeurs onder u worden weggelaten. Dat scheelt natuurlijk geheugenruimte.

De vragen en antwoorden staan in de vorm van DATA-regels in het geheugen. Dat is eenvoudig en overzichtelijk. Het heeft als nadeel, dat er geheugenruimte in beslag wordt genomen. Ook kan een eenmaal gemaakt programma alleen voor een bepaalde toets worden ingezet. Maar dat heeft gelijk ook weer voordelen, zeker voor de leerkracht, die nog niet zo vertrouwd is met het gebruik van de computer in de klas. In andere software uit onze serie maakten we gebruik van een omhulsel, waarin de gegevens van buiten af (cassette of diskette) werden ingeladen. Zo werd die software zeer flexibel in het gebruik. De leerkrachten die dat programma in kwestie al hebben ingetoetst, kunnen wellicht proberen dit multiple choice programma op eenzelfde leest te schoeien. Onmogelijk

is dat in het geheel niet. De eerder gebruikte routines kunnen zo worden overgenomen.

Het programma stap voor stap

Tot en met regel 140 gebeurt er niets ingewikkelds. Het scherm wordt op de gewenste kleur gezet en de keuze van de tekenset valt op kleine- en hoofdletters. Verder wordt het scherm gewist. Deze zaken worden gerealiseerd door middel van PRINT CHR$-opdrachten. Dat werkt prettig en u hoeft de lijst van schermcodes uit de handleiding niet te raadplegen. Dan worden de vragen en antwoorden en nog wat stuurgegevens vanuit de DATA-regels ingelezen. De DATA-regels zelf zijn aan het einde van het programma geplakt. Dat is nodig, want u moet zelf nog enkele DATA-regels vullen. U kunt die regels ongehinderd uitbreiden, zonder met andere programma-regels in conflict te komen. De DATA-regels lopen vanaf 10000.
De eerste twee variabelen N en T worden in het voorbeeldprogramma gevuld met de waarden 2 en 10. Het eerste getal vertegenwoordigt het aantal opdrachten en het tweede getal de bedenktijd per opdracht in seconden. U kunt hierin zelf verandering brengen, als u wilt. In de eerste zal dat zeker moeten, want er mogen geen twee opdrachten zijn met de naam toets. Zorg er wel voor, dat het getal in N overeenkomt met het aantal opdrachten, anders gaat het mis.

Dan worden de eigenlijke vragen samen met de mogelijke antwoorden ingelezen in een tweedimensionale array genaamd Q$(i,j). De variabele i stelt het nummer van de opdracht voor en bij elke i horen 5 waarden van j. In element j=0 zit de vraag, in de elementen j=1 to 4 de vier mogelijke antwoorden. Het nummer van het goede antwoord wordt ingelezen in G(i). Dat verklaart gelijk het formaat van elke DATA-regel. Dat formaat mag u niet zonder meer veranderen. Dus eerst de vraag, dan de vier antwoorden en tot slot het nummer van het juiste antwoord. U kunt zo het aantal vragen vrijwel onbeperkt uitbreiden. Als het gebruik van array's problemen oplevert, raad ik u aan de BASIC-cursus, die momenteel in ons blad wordt geplaatst, eens goed door te lezen. Echt ingewikkeld is het niet.

De schermpresentatie

Omdat de Commodore 64 geen functies (zoals LOCATE) heeft om de cursor op een bepaalde plaats te brengen, gebruiken we weer twee strings voor de horizontale en vertikale cursorverplaatsing. In de vorige afleveringen van deze serie artikelen gebruikten we die techniek ook. String NE$ is voor de regelplaatsing en RE$ is voor de kolomplaatsing. Op regel 300 zien we staan:

300 x=1:y=1:b=36:h=3:gosub 1000

Deze regel is van belang voor de schermpresentatie van het programma. We vinden dat bij een leerprogramma ook de uiterlijke vorm een belangrijke plaats moet innemen. De waarden die worden gezet, worden gebruikt in een subroutine vanaf regel 1000. Die subroutine tovert een van te voren gedefinieerd kader op het scherm. In dat kader komt dan de te stellen vraag. Op deze wijze vestigen we de aandacht van de leerling nadrukkelijk op de vraag. Regel 320 ziet er bijna net zo uit. Het verschil zit in de waarden van de variabelen x,y,b en h. En de subroutine voor een kadertje wordt weer aangeroepen. Nu echter op en geheel andere plaats op het scherm. Dat komt door de getallen behorende bij de genoemde variabelen. In het tweede kader verschijnen de vier mogelijke antwoorden op de vraag. Voor de antwoorden staan de nummers 1 tot en met 4. De leerling hoeft alleen maar op het nummer van het door hem gekozen antwoord te drukken om zijn keuze kenbaar te maken.

Op het moment, dat de vier antwoorden er staan, gaat de beschikbare tijd per vraag in. De subroutine op 1170 zet de klok op nul en er wordt op een toetsindruk gewacht (zie: regel 360). Die toetsindruk wordt vervolgens op juistheid gecontroleerd (regel 510 en verder), de gebruikte tijd wordt vergeleken met de toegestane tijd en het gekozen antwoord wordt invers in het kader geplaatst. Zie daarvoor de regels 550-570. Het programma vertakt zich dan naar regel 590 als het antwoord fout was of de tijdslimiet werd overschreden, met het plaatsen van een kader en de mededeling, dat het antwoord niet goed was. Als wel het goede antwoord werd gegeven, gaat het programma verder op regel 610, waar wederom een kader wordt gezet met daarin een passende mededeling. In beide gevallen vervolgt het programma zijn loop na enkele seconden met de volgende vraag.
Als alle vragen zijn gesteld en beantwoord, krijgt de leerling zijn score in beeld te zien en het verzoek om op RETURN te drukken. Als dat is gedaan start het programma opnieuw met dezelfde reeks vragen.

De kaders

Het plaatsen van de kaders gebeurt door middel van een relatief eenvoudige techniek. Het resultaat is evenwel steeds zeer bevredigend en verhoogt de overzichtelijkheid voor de betrokken leerling. De kaders worden gerealiseerd in de subroutine, die begint op regel 1000.
De variabelen x en y bevatten de schermcoördinaten op de twee assen: x geeft de kolom aan en y de regel op het scherm. b en h bevatten respectievelijk de breedte en de hoogte van het kader. In de subroutine wordt het kader opgebouwd, gepositioneerd en vervolgens op het scherm gezet. Er wordt steeds gewerkt met CHR$ om bepaalde schermtekens uit de tekenset van de Commodore 64 aan te duiden. Aan de hand van de inhoud van b worden er drie elementen op de juiste breedte gebracht, namelijk K1$, K2$ en k3$. Zie daarvoor de regels 1020 tot en met 1050. De eerste is de bovenrand van het kader met hoekjes: CHR$(176) en CHR$(174). De tweede twee rechtopstaande streepjes met daartussen een aantal spaties gelijk aan de inhoud van variabele b. Het derde element is de onderrand, ook met twee hoekjes: CHR$(173) en CHR$(189). Dan is het kader klaar en wordt vervolgens geprint op het scherm. Met delen van de variabelen NE$ en RE$ worden alle elementen op de juiste plaats gezet.

Het gebruik

U kunt dit programma, uiteraard voorzien van uw eigen aanvullingen, zo in de klas gebruiken. Om een vinger aan de pols te houden, zult u steeds aan de leerling die klaar is, de score moeten opvragen, want een administratie is er niet bij. Daar zult u zelf voor moeten zorgen. Eerder gepubliceerde leerprogramma's in deze serie zijn wel voorzien van een administratie. Het leek ons niet zinvol, dit steeds weer opnieuw te doen. Een handige programmeur ziet echter snel de aansluitingsmogelijkheden van alle leerprogrammatuur uit de serie in Commodore-Info. Sommige oude nummers van Commodore-Info zijn nog steeds verkrijgbaar. Elders in dit blad vindt u daarover informatie.
Nog even dit. Elke leerling kan het programma stoppen of in de war sturen. Dat is natuurlijk niet de bedoeling. Om dit te voorkomen, kan de interrupt, die de STOP-toets steeds

controleert, worden uitgezet. U kent de POKE statements wel. Ze staan vaak in de computerbladen. Het nare is, dat bij het gebruik van de meeste van die POKE's ook de tijdklok wordt uitgezet. POKE 808,234 blokkeert de STOP-toets en laat tevens de tijdklok lopen. Gebruik deze dus voor het beveiligen van het programma. Doe dat pas, als het programma zonder fouten loopt. Als de POKE is geaktiveerd, is stoppen, listen of saven niet meer mogelijk.
Heeft u gewerkt met dit leerprogramma (en met de eerder gepubliceerde) en heeft u bepaalde ervaringen, een uitgesproken mening of zinvolle suggesties, laat dat dan eens weten. De redaktie ziet uw brieven graag tegemoet.

B.M.

Listing leerprogramma 5: Multiple Choice

```
100 rem multiple choice
110 :
120 rem instellen scherm
130 poke 53280,0:poke 53281,0
140 print chr$(14)chr$(8)chr$(147)
150 :
160 rem inlezen vragen/antwoorden
170 read n:  rem aantal opdrachten
180 read t:  rem tijd per opdracht
190 for i=1 to n:for j=0 to 4
200 read q$(i,j):next j
210 read g(i):next i
220 :
230 rem schermpositie strings
240 ne$=chr$(19):for i=1 to 20
250 ne$=ne$+chr$(17)
260 re$=re$+chr$(29):next i
270 :
280 rem opstarten
290 for o=1 to n
300 x=1:y=1:b=36:h=3:gosub 1000
310 print left$(re$,2);q$(o,0)
320 x=10:y=10:b=20:h=6:gosub 1020
330 for i=1 to 4
340 print left$(re$,x+2)i;
350 print q$(o,i):next i
360 gosub 1170:gosub 510:next o
370 :
380 rem einde oefening
390 x=7:y=11:b=24:h=10:gosub 1000
400 print left$(re$,x+3);
410 print"De oefening is klaar."
420 print:print:print
430 print left$(re$,x+3);
440 print"Je had"g" goed."
450 x=9:y=19:b=20:h=1:gosub 1020
460 print left$(ne$,y+1)left$(re$,x+2);
470 print"Druk op RETURN"
480 gosub 760:if asc(a$)=13 then run
490 goto 480
500 :
510 rem antwoord halen
520 gosub 760:gosub 1190
530 a=val(a$):if a<1 or a>4 then 520
540 if tt=0 then 590
550 print left$(ne$,a+y+1);
560 print left$(re$,x+5);
570 print chr$(18)q$(o,a)
580 if a=g(o) then gosub 610:return
590 gosub 670:return
600 :
610 rem goed
620 x=1:y=10:b=36:h=6:gosub 1020
630 print left$(ne$,y+2)left$(re$,x+10);
640 print"Dat was goed !!!"
650 g=g+1:for i=1 to 2000:next i:return
660 :
670 rem niet goed
680 x=1:y=17:b=36:h=3:gosub 1020
690 printleft$(ne$,y+1)left$(re$,x+10);
700 print"Dat was fout !!!"
710 print left$(ne$,y+3)left$(re$,x+3);
720 print"Het was: ";q$(o,g(o))
730 for i=1 to 2000:next i:return
740 return
750 :
760 rem toets afwachten
770 get a$:if a$="" then 770
780 return
790 :
1000 rem kader plaatsen
1010 printchr$(147);
1020 k$="":s$="":for i=1 to b
1030 k$=k$+chr$(96)
1040 s$=s$+chr$(32)
1050 next i
1060 k1$=chr$(176)+k$+chr$(174)
1070 k2$=chr$(125)+s$+chr$(125)
1080 k3$=chr$(173)+k$+chr$(189)
1090 print left$(ne$,y);
1100 print left$(re$,x)k1$
1110 for i=1 to h
1120 print left$(re$,x)k2$:next i
1130 print left$(re$,x)k3$
1140 print left$(ne$,y+2);
1150 return
1160 :
1170 rem tijd instellen
1180 ti$="000000":tt=1:return
1190 rem tijd controleren
1200 if ti > t*60 then tt=0
1210 return
1220 :
10000 rem vragen en antwoorden
10010 data 2,10:rem aantal opdrachten/tijd in sec.
10020 data"Welk voorwerp hoort niet in deze rij"
10030 data bloemperk,tuinpad,computer,terras,3
10040 data"Wat is een trog"
10050 data lage drukgebied,etensbak,ziekte,dier,2
```

Als U onze print-out rubriek al heeft bekeken dan is het U zeker opgevallen dat onze listings op een andere manier zijn uitgeprint. We hebben hiervoor een programma gebruikt dat speciaal voor ons is geschreven, **Printlist V1.1**.

Printlist V1.1



Listings om in te lijsten

Het programma dat we voorheen altijd gebruikten, het vertrouwde visilist, is al enige jaren oud. Sindsdien is er heel wat veranderd. er zijn een aantal Commodore computers bijgekomen, daardoor voldoet het programma niet helemaal meer. Nu is het wel mogelijk een programma aan te passen, maar het is veel beter een nieuw programma te schrijven dat volledig up to date is. Dat nieuwe produkt ligt hier voor u.

Wat doet Printlist, hoe moet het gebruikt worden, allemaal vragen die boven komen bij een nieuw programma. Kortweg gezegd: Printlist vertaalt alle tekens die niet of niet duidelijk door een printer zijn af te drukken, naar leesbare tekens. Het programma is volledig menugestuurd.

Voor alle Commodore's

Het sterkste punt van dit programma is dat U op elke Commodore computer listings uit kan draaien van de verschillende Commodore's. Een listing van de oude pet, vic 20, c64, c16, plus 4 en natuurlijk ook de Commodore 128, het kan allemaal zonder problemen. Printlist V1.1 werkt wel alleen met een diskdrive.

Na het runnen van het programma wordt er om de naam van het programma gevraagd dat u wilt gaan uitlisten. Dit kan zowel op het scherm als op de printer (device 4 of 5) gebeuren. De papierlengte staat ingesteld op 72 regels maar kan via het menu naar eigen wens worden aangepast in elke andere lengte.

Standaard staat het programma ingesteld om een listing van een Commodore 64 en vic 20 uit te listen. Wilt u een programma van een andere Commodore computer uitlisten dan niet vergeten het programma voor de juiste computer in te stellen. Dit gaat in het menu eenvoudig door de juiste keuze te maken. Als laatste wordt er gevraagd of alle ingestelde waarden juist zijn, zo niet dan kunt u dit als nog veranderen. Is alles goed ingesteld dan is een druk op de return toets voldoende om het programma in werking te stellen.

Al onze listings zijn met behulp van dit programma uitgeprint. Het is erg eenvoudig om deze listings over te typen. U hoeft maar één ding goed te onthouden, alles wat tussen de grote, rechte haken staat moet uitgevoerd worden. Dus niet letterlijk overnemen.

Bijvoorbeeld:

```
40   w$=w$+"[CRSR-DOWN]
     [CRSR-LEFT]B
     [CRSR-DOWN]c[COM 9]
     A[COM 0]"
```

Hierna typt U in: 40 w$=w$+" dan voert U uit wat tussen de eerste haken staat uit door op de toets te drukken waarmee de cursor naar beneden gaat. Daarna de toets die de cursor naar links stuurt. Dit geeft steeds een revers-teken op het scherm. U vervolgt de listing door het intypen van een hoofdletter B, gevolgd door het indrukken van de toets die zorgt voor het naar beneden gaan van de cursor. Een kleine letter c wordt gevolgd door het indrukken van de combinatie Commodore- en de 9 toets. Een hoofdletter A en als afsluiting de combinatie Commodore en de 0 toets, en de aanhalingstekens. Het belangrijkste is dus alles dat gewoon geschreven staat letterlijk over te nemen, wat in hoofdletters staat ook in hoofdletters overtypen. De kleine letters typt u ook in uw listing met kleine letters. Alles wat tussen de grote rechte haken staat altijd uitvoeren. Deze haken mag U nooit overtypen.

Ook naar PC

Een listing van een Commodore computer die op deze manier is bewerkt is zonder al te veel moeite naar een andere computer over te zetten, bijvoorbeeld naar een PC. Niet dat het programma zo zal werken maar het kan wel veel typewerk besparen. Maar hierop komen we in een andere editie terug. De listing van de Printlist is voor diegene die met een Commodore printer MPS 1000 werken zonder meer over te nemen. Heeft u een andere printer dan moet U in regel 100 vanaf de laatste dubbele punt de tekst weglaten dus :mps1000=1 laat U er helemaal uit.

BLAD 1 Programma printlist v1.1

```
100 z$=chr$(.):print"[SHIFT CLR][2xHOM
    E][CTRL N]":dim tk$(255),ch$(255):
    mps1000=1
101 print"[3xCRSR-RIGHT]PrintList[SPAC
    E]v1.1[2xSPACE]--[SPACE](C)87[SPAC
    E]HSP":print"[CRSR-DOWN][3xCRSR-RI
    GHT]Een[SPACE]momentje[SPACE]graag
    [SPACE]";
102 print".";:fori=128to232:readtk$(i)
    :next:rem tokens basic 2/4      $80
    -$e8
103 print".";:fori=076to125:readtk$(i)
    :next:rem tokens basic 3.5      $cc
    -$fd
104 print".";:fori=066to074:readtk$(i)
    :next:rem tokens basic 7.0 $ce $02
    -$0a
105 print".";:fori=002to038:readtk$(i)
    :next:rem tokens basic 7.0 $fe $02
    -$26
106 print".";:fori=001to032:readch$(i)
    :next:rem  quotecodes
107 print".";:fori=129to192:readch$(i)
    :next
108 print".";:fori=219to223:readch$(i)
    :next:readch$(255)
109 print"[HOME][3xCRSR-DOWN][3xCRSR-R
    IGHT]Programmanaam[3xSPACE]*[9xSPA
    CE][12xCRSR-LEFT]";
110 inputf$:f$=left$(f$,16):iff$=""the
    n109
111 dd=8:print"[HOME][5xCRSR-DOWN][5xC
    RSR-RIGHT]Disk[SPACE]DEVICE[2xSPAC
    E]"str$(dd)"[3xCRSR-LEFT]";
112 inputd$:dd=val(d$):ifdd<8ordd>15th
    en111
113 pd=4:print"[HOME][6xCRSR-DOWN][4xC
    RSR-RIGHT]Print[SPACE]DEVICE[2xSPA
    CE]"str$(pd)"[3xCRSR-LEFT]";
114 inputd$:pd=val(d$):ifpd<3orpd>6the
    n113
115 pl=72+47*(pd=3):print"[HOME][7xCRS
    R-DOWN][5xCRSR-RIGHT]Blad[SPACE]LE
    NGTE[2xSPACE]"str$(pl)"[4xCRSR-LEF
    T]";
116 inputd$:pl=val(d$):ifpl<.then115
117 ll=40+(pd=3):print"[HOME][8xCRSR-D
    OWN][4xCRSR-RIGHT]Regel[SPACE]LENG
    TE[2xSPACE]"str$(ll)"[4xCRSR-LEFT]
    ";
118 inputd$:ll=val(d$):ifll<20orll>160
    then115
119 print"[HOME][10xCRSR-DOWN]","MODUL
    E[SHIFT SPACE]1[SPACE]=[SPACE]PET,
    VIC20,CBM64
120 print,"MODULE[SHIFT SPACE]2[SPACE]
    =[SPACE]CBM16[SPACE]en[SPACE]PLUS/
    4":print,"MODULE[SPACE]3[SPACE]=[S
    PACE]COMMODORE[SPACE]128
121 pm=1:print"[HOME][14xCRSR-DOWN][4x
    CRSR-RIGHT]Print[SPACE]MODULE[2xSP
    ACE]"str$(pm)"[3xCRSR-LEFT]";
122 inputd$:pm=val(d$):ifpm<1orpm>3the
    n121
123 print"[HOME][16xCRSR-DOWN]","[4xCR
    SR-RIGHT]OK[3xSPACE]j[3xCRSR-LEFT]
    ";
124 inputd$:ifleft$(d$,1)="n"thenrun
125 open15,dd,15:open8,dd,8,f$+",p,r":
    input#15,en,em$,et,es:ifen=.then12
    7
126 print"[SHIFT CLR]"em$:close8:close
    15:end
127 get#8,x$:get#8,x$:ifmp=.thenopen4,
    pd,7
128 ifmp=1thenopen4,pd:print#4,chr$(27
    )chr$(120)"1";
129 gosub153
130 get#8,l1$:get#8,l2$:ifasc(l1$+z$)=
    .andasc(l2$+z$)=.then144
131 get#8,l$:x=asc(l$+z$):get#8,l$:r=2
    56*asc(l$+z$)+x:q=.
132 p$=right$("[4xSPACE]"+mid$(str$(r)
    ,2)+"[SPACE]",6)
133 get#8,c$:c=asc(c$+z$):onqgoto138:i
    fc=34thenq=1
134 ifc=.thengosub147:goto130
135 if(c=254orc=206)andpm=3thence=-(c=
    206):fe=-(c=254):goto133
136 ifc<128andce=.andfe=.thenp$=p$+c$:
    gosub146:goto133
137 p$=p$+tk$(c-128*(c>203)*(pm>1)-64*
    (fe=1)):fe=.:ce=.:gosub146:goto133
138 ifct=.thenq=.:on-(c=.orc=34)goto13
    4:q=1:a=c:h$=chr$(c):ct=1:goto133
139 ifa=cthenh$=h$+chr$(c):ct=ct+1:got
    o133
140 ifch$(a)<>""thenh$=ch$(a):ifct=1th
    enh$="["+h$+"]"
141 ifch$(a)<>""andct>1thenh$="["+mid$
    (str$(ct),2)+"x"+h$+"]"
142 p$=p$+h$:gosub146:ct=.:ifc=.orc=34
    thenq=.:goto134
143 goto138
144 close8:close15:print#4:l=l+1
145 p$="[6xSPACE]**[SPACE]EINDE[SPACE]
    LISTING[SPACE]"+f$:gosub148:print#
    4:close4:end
146 iflen(p$)<llthenreturn
147 ifc=0andlen(p$)=6then149
148 l=l+1:pc$=left$(p$,ll):gosub155:p$
    ="[6xSPACE]"+mid$(p$,ll+1)
149 ifl<pl-3thenreturn
150 ifpd<>3thenfori=ltopl:print#4:next
    :goto153
151 print"[CRSR-DOWN][6xSPACE][CTRL 9]
    [SPACE]druk[SPACE]nu[SPACE]op[SPAC
    E]een[SPACE]toets[SPACE]"
152 getg$:ifg$=""then152
153 print"[SHIFT CLR]";:pg=pg+1:print#
    4
154 pc$="[6xSPACE]BLAD"+str$(pg)+"[SPA
    CE]Programma[SPACE]"+f$:gosub155:p
    rint#4:l=4:return
155 ifpd=3ormp=.thenprint#4,pc$:return
156 pd$=pc$:pc$="":fori=1tolen(pd$):a=
    asc(mid$(pd$,i))
157 pc$=pc$+chr$(a+32*(a>64)*(a<91)-12
    8*(a>192)*(a<219)-189*(a=48)):next
158 print#4,pc$chr$(27)"j"z$pc$:return
159 data end,for,next,data,input#,inpu
    t,dim,read
160 data let,goto,run,if,restore,gosub
    ,return,rem
161 data stop,on,wait,load,save,verify
    ,def,poke
```

```
162 data print#,print,cont,list,clr,cm
    d,sys,open
163 data close,get,new,tab(,to,fn,spc(
    ,then
164 data not,step,+,-,*,/,↑,and
165 data or,>,=,<,sgn,int,abs,usr
166 data fre,pos,sqr,rnd,log,exp,cos,s
    in
167 data tan,atn,peek,len,str$,val,asc
    ,chr$
168 data left$,right$,mid$,go,concat,d
    open,dclose,record
169 data header,collect,backup,copy,ap
    pend,dsave,dload,catalog
170 data rename,scratch,directory,dcle
    ar,bank,bload,bsave,key
171 data delete,else,trap,resume,dispo
    se,pudef,using,err$
172 data instr,rgr,rclr,rlum,joy
173 data rdot,dec,hex$,err$,instr,else
    ,resume,trap
174 data tron,troff,sound,vol,auto,pud
    ef,graphic,paint
175 data char,box,circle,gshape,shape,
    draw,locate,color
176 data scnclr,scale,help,do,loop,exi
    t,directory,dsave
177 data dload,header,scratch,collect,
    copy,rename,backup,delete
178 data renumber,key,monitor,using,un
    til,while
179 data pot,bump,pen,rsppos,rsprite,r
    spcolor,xor,rwindow,pointer
180 data bank,filter,play,tempo,movesp
    r,sprite
181 data sprcolor,rreg,envelope,sleep,
    catalog,dopen,append,dclose
182 data bsave,bload,record,concat,dve
    rify,dclear,sprsave,collision
183 data begin,bend,window,boot,with,s
    prdef,quit,stash
184 data "[SPACE]",fetch,"[SPACE]",swa
    p,off,fast,slow
185 data "CTRL[SPACE]A","CTRL[SPACE]B"
    ,"CTRL[SPACE]C","CTRL[SPACE]D","CT
    RL[SPACE]2","CTRL[SPACE]F","CTRL[S
    PACE]G"
186 data "CTRL[SPACE]H","CTRL[SPACE]I"
    ,"CTRL[SPACE]J","CTRL[SPACE]K","CT
    RL[SPACE]L","","CTRL[SPACE]N","CTR
    L[SPACE]O"
187 data "CTRL[SPACE]P","CRSR-DOWN","C
    TRL[SPACE]9","HOME","DEL","CTRL[SP
    ACE]U","CTRL[SPACE]V"
188 data "CTRL[SPACE]W","CTRL[SPACE]K"
    ,"CTRL[SPACE]Y","CTRL[SPACE]Z","CT
    RL[SPACE][","CTRL[SPACE]3"
189 data "CRSR-RIGHT","CTRL[SPACE]S","
    CTRL[SPACE]7","SPACE"
190 data "COM[SPACE]1",,,,"F1","F3","F
    5","F7","F2","F4","F6","F8","","",
    ""
191 data "CTRL[SPACE]1","CRSR-UP","CTR
    L[SPACE]0","SHIFT[SPACE]CLR","SHIF
    T[SPACE]INST"
192 data "COM[SPACE]2","COM[SPACE]3","
    COM[SPACE]4","COM[SPACE]5","COM[SP
    ACE]6","COM[SPACE]7","COM[SPACE]8"
193 data "CTRL[SPACE]5","CRSR-LEFT","C
    TRL[SPACE]8","CTRL[SPACE]4","SHIFT
    [SPACE]SPACE"
194 data "COM[SPACE]K","COM[SPACE]I","
    COM[SPACE]T","COM[SPACE]@","COM[SP
    ACE]G","COM[SPACE]+","COM[SPACE]M"
195 data "COM[SPACE]£","SHIFT[SPACE]£"
    ,"COM[SPACE]N","COM[SPACE]Q","COM[
    SPACE]D","COM[SPACE]Z","COM[SPACE]
    S"
196 data "COM[SPACE]P","COM[SPACE]A","
    COM[SPACE]E","COM[SHIFT SPACE]R","
    COM[SPACE]W","COM[SPACE]H","COM[SP
    ACE]J"
197 data "COM[SPACE]L","COM[SPACE]Y","
    COM[SPACE]U","COM[SHIFT SPACE]O","
    SHIFT[SPACE]@","COM[SPACE]F","COM[
    SPACE]C"
198 data "COM[SPACE]X","COM[SPACE]V","
    COM[SPACE]B","SHIFT[SPACE]*"
199 data "SHIFT[SPACE]+","COM[SPACE]-"
    ,"SHIFT[SPACE]-","COM[SPACE]PI","C
    OM[SPACE]*","COM[SPACE]PI"

** EINDE LISTING printlist v1.1
```



Luc Sala's Datakolom

Hi-tech Isolationisme

De handelsoorlog tussen Japan en de VS begint dreigende vormen aan te nemen, aan beide zijden van de Stille Oceaan roepen de "Hard-liners" om nog meer beperkingen, tariefmuren en vergeldingsmaatregelen. Ook Europa krijgt hier mee te maken, al was het maar omdat de prijs van de RAM-chips omhoog gaat en daardoor de consument uiteindelijk deze prestige-strijd betaalt.

In Amerika is de kreet "Buy American" een van de beste verkoopargumenten aan het worden, zeker in de computerhandel. Dat ook van de grote merken (o.a. IBM) een groot deel van de produktie uit het verre Oosten komt, wordt gemakshalve maar vergeten. Reagan heeft in een soort anti-Japanse handelskruistocht nu o.a. de Japanse laptop computers en de RAM geheugenchips met een 100% tarief belast. Dit leidt ook voor Europa tot een schaarste aan RAM-chips en sterk verhoogde prijzen, terwijl de verdere ontwikkeling van de chipstechnology en de afzetpatronen van bijvoorbeeld de 1 en 4 MegaBit chips daardoor in gevaar komt.

Zoals we in ons land al vele eeuwen weten, is vrije handel de basis van een groot deel van onze welvaart. En dat was sinds 1945 ook internationaal een duidelijke zaak, geleidelijk moesten alle belemmeringen voor vrije handel worden afgebroken. Handel drijven is de beste tegenmaatregel tegen oorlogen. Het uitwisselen van produkten en diensten, en mede daardoor een betere communicatie tussen de handelspartners is ook de basis van de EEG gedachte.
Daarom is het zo verwonderlijk, dat er nu tegenover Japan een soort ommezwaai aan de gang is. Vrijwel iedereen is tegen handelsbelemmeringen tussen de Europese landen, tussen de VS en Europa, maar tegenover Japan is er een andere stemming, en vindt men tariefmuren of importquota wel zinvol.

Misschien is een goede buur beter als een verre vriend, maar ik vindt het is toch wel kortzichtig om nu mee te doen in de eenzijdige en veelal weinig doordachte maatregelen, die Reagan heeft afgekondigd tegenover Japan. Want die maatregelen gaan voorbij aan wat de wezenlijke problemen zijn, houden geen rekening met de totaal verschillende handelsculturele opvattingen van Japan en andere landen in het Verre Oosten, en zijn in hun uitwerking vaak ook nog contra-productief. De stopzetting van bijvoorbeeld de export van de Toshiba 3100 laptops naar Amerika heeft alleen maar geleid tot een verhoogde inspanning van Toshiba in Japan zelf en in Europa. Iedere 3100 computer, die niet in de VS verkocht wordt, gaat nu vast een koper vinden in Europa. En dat betekent, dat men veel moeite zal krijgen om bijvoorbeeld de Compaq, Zenith en Kaypro portables hier te slijten.

Het is niet logisch om te denken, dat een tariefmuur leidt tot het zo maar stopzetten van een hele produktlijn. Het gebruiken van dit soort maatregelen als een soort dwangmiddel om concessies over de hele linie van Japan te krijgen gaat voorbij aan de op dit moment vrij precaire economische situatie in Japan zelf, aan de manier waarop men zaken doet met het land en aan de lessen die men uit de historie zou kunnen leren.
Volgens nogal wat historici is bijvoorbeeld de oorlog tussen Japan en de VS (WO II) het gevolg geweest van enorme miscommunicatie tussen beide landen en van het onjuist interpreteren van (mede) economische conflicten.
Nu gaat het wat ver om te zeggen, dat de handelsoorlog tussen de VS en Japan zou leiden tot een echte oorlog, maar de stemming in zowel de VS als in Japan is over dit onderwerp bepaald niet optimistisch. Men vreest verdere escalatie. Gezien de precaire economische situatie zijn de politici geneigd om de schuld daarvan maar overzee te zoeken en wordt het publiek opgehitst met "Buy American" campagnes, natuurlijk weer gevolgd door soortgelijke acties in Japan. De bedrijven, die hier een snel voordeeltje zien, zoals de chipsmakers in de VS, haasten zich om profijt te slaan uit deze stemming en druk uit te oefenen op de politici. Dat isolationisme op technisch gebied vrijwel zeker een doodlopende weg is, vergeten ze gemakshalve maar even, in de VS zijn de resultaten van het volgende kwartaal altijd veel belangrijker dan de ontwikkelingen op lange ter-



mijn. Houdt de aandeelhouders tevreden, anders gaan de koersen omlaag.

Aan de andere kant wordt nu ook in Japan, waar de economische situatie verre van rooskleurig is, een anti-westerse stemming aangewakkerd. Zo lang je de schuld voor de economische crisis maar ergens anders kunt leggen, blijven de nationale politici buiten schot.
De vraag is nu of we aan deze ontwikkeling iets kunnen doen. Natuurlijk is dit een politieke zaak, maar het aantal politici in ons land of zelfs op Europees niveau, die iets meer van Japan (willen) weten dan het leren eten met stokjes voor een leuk werkbezoekje, is zeer gering. Men weet er weinig van, het is te ver weg, te vreemd voor een comfortabel reisje, en het land ligt ook niet zo lekker in de publieke opinie, mede vanwege WO II. Het idee, dat de Japanners de hele wereld ondersneeuwen met hun produkten en de werkgelegenheid hier in gevaar brengen, is de afgelopen decennia voldoende rondgebazuind. De politici zullen eerder geneigd zijn, te kiezen voor meer sancties, meer beperkingen en dus voor een verdere agressiespiraal.

Is dan het bedrijfsleven geroepen om wat verder te kijken en iets te doen aan de vermindering van de spanning? Zij lijken het meest gebaat bij de sancties, maar komen op lange termijn ook het meest in de knel, dus zouden zij ook de mogelijke escalatie moeten voorkomen. Nu is er vanuit ons land natuurlijk best wel een redelijke handel met Japan, maar helaas blijft dat veelal beperkt tot de grote bedrijven.
De rol van Philips, met toch wel grote belangen en zeer goede contacten in Japan, maar ook een beladen handelsverleden in dit opzicht, is bijna tweeslachtig te noemen. Enerzijds is dat al zeer lang samen met de Japanners streven naar bepaalde standaards, werken in joint-ventures en andere samenwerkingsverbanden. Anderzijds (met nogal wat gewroet en gerommel) de pogingen om de Japanners uit bepaalde Europese markten te houden, met name die voor TV's en VCR's. Maar de Japanners weten langzamerhand, dat men in Eindhoven wel degelijk op de lange termijn werkt en vooral een open relatie met het land en de bedrijven daar belangrijk acht. Er wordt wel wat gestoeid, maar het blijft een gevecht onder heren, Wisse Dekker weet dat er geregeld een glaasje sake nodig is om de zaken met Morita en Matsushita bespreekbaar te houden.

Dat begrijpen de Amerikanen veel minder, die sturen af en toe een afgezant, maar die slaakt alleen maar dreigende taal en reist dan snel weer af. In Japan werkt dat niet, daar moet je steeds maar weer terugkomen, blijven praten, blijven glimlachen en eerst ja en amen zeggen, voordat je voorzichtig op mogelijk voor beiden interessante alternatieven wijst en zo je eigen mening tactisch naar voren brengt.

Het zou goed zijn, wanneer meer kleinere bedrijven handel zouden drijven met Japan, omdat daarmee het aantal contacten tussen de landen snel toeneemt. Kleinere bedrijven zullen eerder geneigd zijn zowel naar import als export te kijken en door meer gespreide handelspatronen overheidsbemoeiing en sancties moeilijker te maken. Het is nu eenmaal eenvoudiger om vijf grote chipsmakers te treffen met een maatregel dan bijvoorbeeld de kleding- en modewereld.

Er zijn nogal wat organisaties, die zich bezighouden met de promotie van de handel. Helaas is het zo, dat die meestal gericht zijn op het stimuleren van de handel in één richting. Vanuit Japan zijn het MITI en Jetro acvtief, vanuit Nederland zijn er natuurlijk de Kamer van Koophandel en de diverse exportbevorderende instanties zoals de EVD. Sinds een paar jaar zijn een aantal importeurs van Japanse produkten (DUJAT) ook bezig met de promotie van Nederlandse produkten in Japan, ze hebben ingezien dat een evenwichtige wederzijdse handel uiteindelijk in ieders belang is.

Gezien de reëel dreigende situatie op handelsgebied, waar misschien niet helemaal toevallig nu de computer-industrie in het brandpunt van de belangstelling staat, is echter een verdere stimulatie van de handel met Japan dringend gewenst.

Luc Sala

●

GEOS HOT News

Op de CES in Chicago was er van de kant van Berkeley Softworks wel weer veel nieuws. Men is druk bezig, om een hele Geos ontwikkelomgeving te maken. Niet alleen met nieuwe toepassingen als Desktop Publishing, maar ook met programmeerhulpen als GeoProgrammer. En de nieuwe versie van Geos, namelijk V1.3., is weer wat sneller en gebruiksvriendelijker. Vooral op het gebied van de printerinterfaces is er een stevige uitbreiding, o.a. met de Apple laserwriter voor PostScript.

Het meest interessant was wel *GeoPublish*, een nieuw GEOS pakket om aan Desktop Publishing te kunnen doen. Op een 64 is dat een hele prestatie, iedereen die met Newsroom of PrintShop gewerkt heeft, weet tegen welke systeembeperkingen men aanloopt. De speciale eigenschappen van Geos, zoals de vrijheid om eigen fonts op het scherm te gebruiken, wordt hier echter ten volle benut, waardoor GeoPublish veel meer biedt. Het mengen van plaatjes en tekst is mogelijk, maar ook het gebruik van tamelijk grote letters voor kopregels (tot 48 punten), vergroten en verkleinen van grafische afbeelding en dergelijke. is mogelijk. Hier is natuurlijk de aansturing van de Apple Laserwriter een belangrijk punt, om wat men in beeld krijgt ook perfect te kunnen afdrukken. GeoPublish gaat 70 dollar kosten in de VS.

Met *GeoProgrammer* kan men in machinetaal programma's maken, die dan in Geos passen, dat wil zeggen vanuit Geos kunnen worden opgestart. Dit betekent dus, dat men zelf Geos applicatieprogramma's kan gaan maken. In de praktijk zal men waarschijnlijk eerder bestaande software gaan aanpassen voor Geos en we denken, dat daardoor het Geos aanbod nog sneller zal groeien. In GeoProgrammer zit een *Geolinker* en een *GeoDebugger*.